An Integrated Security Protocol Communication Scheme for Internet of Things using the Locator/ID Separation Protocol Network

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy (PhD)

Faculty of Science and Technology
Middlesex University
United Kingdom

By

Ali Hussein Raheem

Supervised by

Prof. **Aboubaker Lasebae**
Dr. **Mahdi Aiash**
Prof. **Jonathan Loo**

January, 2017

Abstract

Internet of Things communication is mainly based on a machine-to-machine pattern, where devices are globally addressed and identified. However, as the number of connected devices increase, the burdens on the network infrastructure increase as well. The major challenges are the size of the routing tables and the efficiency of the current routing protocols in the Internet backbone. To address these problems, an Internet Engineering Task Force (IETF) working group, along with the research group at Cisco, are still working on the Locator/ID Separation Protocol as a routing architecture that can provide new semantics for the IP addressing, to simplify routing operations and improve scalability in the future of the Internet such as the Internet of Things. Nonetheless, The Locator/ID Separation Protocol is still at an early stage of implementation and the security Protocol e.g. Internet Protocol Security (IPSec), in particular, is still in its infancy.

Based on this, three scenarios were considered: Firstly, in the initial stage, each Locator/ID Separation Protocol-capable router needs to register with a Map-Server. This is known as the Registration Stage. Nevertheless, this stage is vulnerable to masquerading and content poisoning attacks. Secondly, the addresses resolving stage, in the Locator/ID Separation Protocol the Map Server (MS) accepts Map-Request from Ingress Tunnel Routers and Egress Tunnel Routers. These routers in trun look up the database and return the requested mapping to the endpoint user. However, this stage lacks data confidentiality and mutual authentication. Furthermore, the Locator/ID Separation Protocol limits the efficiency of the security protocol which works against redirecting the data or acting as fake routers. Thirdly, As a result of the vast increase in the different Internet of Things devices, the interconnected links between these devices increase vastly as well. Thus, the communication between the devices can be easily exposed to disclosures by attackers such as Man in the Middle Attacks (MitM) and Denial of Service Attack (DoS).

This research provided a comprehensive study for Communication and Mobility in the Internet of Things as well as the taxonomy of different security protocols. It went on to investigate the security threats and vulnerabilities of Locator/ID Separation Protocol using X.805 framework standard. Then three Security protocols were provided to secure the exchanged transitions of communication in Locator/ID Separation Protocol. The first security protocol had been implemented to secure the Registration stage of Locator/ID separation using ID/Based cryptography method. The second security protocol was implemented to address the Resolving stage in the Locator/ID Separation Protocol between the Ingress Tunnel Router and Egress Tunnel Router using Challenge-Response authentication and Key Agreement technique. Where, the third security protocol had been proposed, analysed and evaluated for the Internet of Things communication devices. This protocol was based on the authentication and the group key agreement via using the El-Gamal concept. The developed protocols set an interface between each level of the phase to achieve security refinement architecture to Internet of Things based on Locator/ID Separation Protocol. These protocols were verified using Automated Validation Internet Security Protocol and Applications (AVISPA) which is a push button tool for the automated validation of security protocols and achieved results demonstrating that they do not have any security flaws. Finally, a performance analysis of security refinement protocol analysis and an evaluation were conducted using Contiki and Cooja simulation tool. The results of the performance analysis showed that the security refinement was highly scalable and the memory was quite efficient as it needed only 72 bytes of memory to store the keys in the Wireless Sensor Network (WSN) device.

To my parents, for all their love, kindness and sacrifice,

Acknowledgments

The quest for a PhD has indeed been a long one! I thank Allah first for His abundant blessings, and for blessing me in particular with the opportunity to pursue a doctorate. I thank Him for supporting me in the face of challenges with His Mercy and Guidance, and for facilitating the completion of this thesis.

First of all, I would like to thank my Director of Studies Prof. Aboubaker Lasebae and Supervisors Dr. Mahdi Aiash and Prof. Jonathan Loo for their invaluable guidance, encouragement, cooperation and support over the years. I thank Middlesex University for giving me the opportunity to do PhD and be one of its students.

Among family and friends, I would like to thank my parents (Prof. Hussein Raheem Mustafa and Prof. Ibtisam Jassm Al-Dourie) for their love and for being very supportive in every way possible. I would like to thank them in particular for ensuring that I keep smiling even during the tough times I had. Their faith in my ability and their constant encouragement to explore new horizons helped bring out the best in me and I remain indebted to them for this.

As always, it is impossible to mention all those who had an impact on this work, however there are those whose spiritual support is even more important. I feel a deep sense of gratitude to my sister Dr. Rand Hussein Raheem, who formed part of my vision and taught me good things that really matter in life. Her infallible love and support have always been my strength. Her patience and sacrifice will remain my inspiration throughout my life. I am also very much grateful to all my family members for their constant inspiration and encouragement.

Author’s Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in text and acknowledgments.

Ali Hussein Raheem

January 2017

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AVISPA	Automated Validation Internet Security Protocol and Applications
AES	Advanced Encryption Standard
ACLs	Access Control
AODV	Ad hoc On Demand Distance Vector Protocol
CoAP	The Constrained Application Protocol
CL-AtSe	Constraint Logic based Attack Searcher
CRA	Challenge-Response Authentication
CA	Certificate Authority
CBC	Cipher Block Chaining
DNS	Domain Name System
DoS	Denial of Service Attack
DTLS	Datagram Transport Layer Security
DHP	Diffie Hellman assumption
DSA	Digital Signature Algorithm
EID	Endpoint IDs
ETR	Egress Tunnel Router
ECC	Eliptic Cure Cryptosystem
E2E	End to End Security
FFD	Full Function Device
HTTP	Hypertext Transfer Protocol
HLPSL	High level Protocols Specification Language
HI	Host Identifier
HIP	Host Identity Protocol
IoT	Internet of Things
IPv6	Internet Protocol Version 6
IETF	Internet Engineering Task Force
ITR	Ingress Tunnel Router
IPsec	Internet Protocol Security
IBC	ID/Based Cryptography
IF	Intermediate Format
IKE	Internet Key Exchange

LISP	Locator/ID separation Protocol
LoWPAN	Low-Power Wireless Personal Area Network
MS	Map Server
MSN	Mobile Sensor Node
MitMA	Man in the Middle Attack
MD5	Message Digest Algorithm
M2M	Machine-to-Machine
MAC	Message Authentication Code
OFMC	On-the Fly Model Checker
PETR	Proxy ETR
PIETR	Proxy ITR
PKI	Public key infrastructure
RLoC	Routing Locators
RR	Return Reputability
RFD	Reduced Functionality Device
RSA	Rivest Shamir Adleman algorithm
SATMC	SAT-based Model-Checker
SA	Security Association
TLS	Transport Layer Security
TA4SP	Tree Automata-based Protocol Analyser
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WSN	Wireless Sensor Node
XTR	Refers to a device which functions both as an ITR and ETR

List of Publications

[1] Ali Raheem, Aboubaker Lasebae, Mahdi Aiash, and Jonathan Loo '**Supporting Communication in the the IoT using the Location/ID Split Protocol: A Security Analysis**' Future Generation Communication Technology (FGCT), 2013 Second International Conference, IEEE, London , 12-14 November, 2013. **(Presented in November-2013)**

[2] Ali Raheem, Aboubaker Lasebae, and Jonathan Loo **' A secure Authentication Protocol for IP-Based Wireless Sensor Communications using the Location/ID split Protocol (LISP)'**The 13th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Beijing, China 24-26 September, 2014. **(Presented in September-2014).**

Chapter 1

Introduction

Internet of Things (IoT) is a network of globally identifiable physical objects (or Things) in their integration with Internet and their representation in the virtual or digital world. In order to build the IoTs, a wide range of technologies are involved such as, the Radio-Frequency Identification (RFID) for location and device identification, improved personal and web area networking protocols, web technologies [Atzori et al., 2010]. These technologies help to build a virtual world of things on top of physical world where things through Machine-to-Machine (M2M) communication talk to each other, through humans-to-machine interactions providing information to humans or taking actions on human inputs, or acting as passive entities to provide data to intelligent entities.

IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) standard is IEEE 802.15.4; it allows the efficient use of the Internet Protocol version 6 (IPv6) over a low power and low rate wireless network on simple embedded devices through an adaptation layer and the optimization of released protocols [kim, 2008]. Adding to this, Wireless Sensor Network (WSN) is a technology that connects the virtual world and physical world where nodes can autonomously communicate among each other and with intelligent systems [Fei et al., 2016]. The Locator/ID Separation Protocol (LISP) is a routing architecture that provides new semantics for IP addressing, in order to simplify routing operations and improve scalability in the future of the Internet such as the IoT [Cisco-A, 2014]. Based on this, the thesis focuses on the IoT formed through the interconnection of IP-connected WSN based on the LISP network architecture.

A conventional WSN is a network of sensor devices that senses and collects environmental data and cooperatively forward it to the router for further processing. However, these first generations WSN lack any standardization support and are mostly used for environmental monitoring and battlefields [Li et al., 2016]. Current WSNs are deployed in environments more close to humans and are aimed for applications such as building automation, bridge and tunnel monitoring, industrial automation and control and human sensing. The sink in current WSN such as LISP routers, i.e., ITR and ETR, can query data from sensor nodes and/or send control message to them [Kafle et al., 2010]. In other words, the sensor nodes are resource-constrained devices with limited storage and processing capabilities because its battery power is connected through weak links.

1.1 Security Cryptography for the Internet of Things

More and more Internet of Things (IoT) devices will be connected to the network [Atzori et al. 2010]. The sensors distributed in the smart city will be able to communicate to another one with smartphones. Therefore, this channel needs security protocol which connect all these devices to the Internet [Cheng et al. 2017]. In order to establish this secure communication in the IoT, cryptography is used as a security measure.

The encryption methods for 6LoWPAN/IoT need to be developed more in order to be adapted to the prevailing constraints in 6LoWPAN/IoT such power and low computing ability. This means that the un-optimised cryptography mechanisms will consume more resources and, therefore, shorten network life time [Ferguson et al. 2010]. Added to this ,the key used in encryption methods should not be too short, otherwise, it will be easy to be broken by any attacks. As 6LoWPAN/IoT is the combination of WSN and the Internet, it is natural to apply these two network cryptography mechanisms for securing this network. To secure the link layer with several operations, WSN uses Advanced Encryption Standard (AES) [Biryukoy et al 2010] modes. Most of these modes do not ensure integrity function. In order to protect network layer end-to end security, IPsec (Internet protocol Security) is utilised with transport and tunnel models. Formerly, the public key cryptography mechanism was thought to be too heavy for applying in WSN. Fortunately, recent developments [Ayuso et al. 2010] show a way to combine Rivest- Shamair–Adelman asymmetric encryption (RSA) and Elliptic Curve Cryptography (ECC) techniques with several modes that can be adjusted to network scenarios. Another problem that should be considered is the exchanging key. The Internet Key Exchange from Internet Protocol Security (IPsec) is suggested for exchanging the key in the network [kundu et al. 2010]. However, the Internet Key Exchange (IKE) is not considered as a feasible solution because of its heavy signalling messages, which are unsuitable for the small packet size of 802.15.4 nature and the energy efficiency requirement. Besides, they lack scalable ability. Therefore, it is very necessary to analyse the threat towards the key at the bootstrap time when an adversary sits among other nodes without being required to be authenticated.

Therefore, this research study used El-Gamal encryption which is lierally an asymmetric key algorithm for public key cryptography which is based on Diffie-Hellman key (DH) exchange.

Here, the public key of B is g^b and the computed DH key g^{ab} are used as one-time pad to encrypt a message $m \in \mathbb{Z}_p^*$ which are a group element of the respective group used, typically the encryption operation is defined as multiplying the message with the DH key or xoring the message with a hash of the DH key [Mikhail et al.2014]. The cipher text is then a tuple (c_1, c_2) consisting of the message encrypted with DH key $m.g^{ab}$ and the part g^a of the DH key computed by the encrypting party. But it should be noted that the entire process is conducted by one party, i.e. the party encrypting the message. This party then sends the tuple $(c_1, c_2) = (g^a, m.g^{ab})$ to the receiver B. Consequently, the advantages of useing El-Gamal in this research are the following [Mikhail et al 2014]:

- It is homomorphic encryption scheme that allows multiplying plaintext hidden inside the cipher texts. When using the homomorphic property with an encryption of the identity element 1 of the group, it allows publicly re-randomising El-Gamal cipher text, i.e. obtaining new cipher texts for the same message which are unlikable to the original cipher text. Furthermore, using exponential El-Gamal obtained from EL-Gamal by encoding the message m as g^m, i.e. as exponent of the generator g, El-Gamal can also be made additively homomorphic for polynomial sized message spaces (since decrypting involves computing discrete logarithms)

- There are efficient honest-verifier zero-knowledge proofs of knowledge to prove properties of El-Gamal cipher texts without revealing the plaintext, e.g., equality of plaintexts.
- EL-Gamal schemes provide a good security system that can exchange and compute the shared secret keys between the devices
- El-Gamal schemes provide End-to-End Security encryption where only the communicating users can read the messages.

1.2 Research Motivations

The IoT has become a ubiquitous term to describe the tens of billions of devices that have sensing or actuation capabilities and are connected to each other via the Internet. This massive increase has a direct effect on the network infrastructure and on the size of the routing table's efficiency of the current routing protocols in the IoT. To address this problem, an IETF working group, along with the research group at Cisco, are working on LISP as a routing architecture that provides new semantics for IP addressing in order to simplify routing operations and improve scalability in the future of the Internet such as the IoT [Chen et al 2016]. However, LISP is still at an early stage of implementation and the security protocol, in particular, is still in its beginning. Therefore, the main motivation of this research is divided into four main parts:

- **Part I:** Investigating and analysing the level of security performance for the IoT based on LISP architecture and exposing its security vulnerabilities using X.805 security framework architecture.
- **Part II:** Modelling the security threats for the IoT based on LISP using Automated Validation Internet Security Protocol and Applications (AVISPA) tool in order to design efficient protocols against these threats.
- **Part III:** Designing and simulating three security protocols namely the registration stage, the resolving stage and the communication procedure for the IoT based on LISP network architecture.
- **Part IV:** Implementing the analysis and the verification of the designed protocols via AVISPA.

1.3 Research Challenges

There has been a tremendous increase in the use of the IoTs, from the 365 million of users in 2000 to 50 billion devices in 2020 [Gartner, 2013]. Nevertheless, this rising growth faces serious challenges related to scalability, manageability, addressing/identity and robustness [Sundmaeker et al., 2010]. Furthermore, the distinct features of the openness and ubiquity of the current Internet are considered in fact a real problem as they do not offer suitable support for privacy and secure communication/mobility. Henceforth, different challenges arise that actually need to be faced in this research especially because this technology has unique characteristics such as low energy combustion, short size memory and small packet size. These challenges were tackled in the IoT as the following:

- Communication/Mobility support in IP-WSN increases the fault tolerance capacity and connectivity, allowing extending and adapting network to change its location and infrastructure. These features are necessary to satisfy the dependability and scalability of the networks of the future world. Several solutions have been developed to support mobility, but actually they still present limitations mainly caused by the role of IP address, as both node ID for session determination in the application/transport layer, and Locator in the network layer [Cisco-A, 2014]. For that reason, this research study is based on one of the first LISP that supports IP-WSN, which has defined compressed and size optimised mobility signalling. The mentioned approach presents the first challenge, since LISP messages are potentially dangerous. For example, a malicious host might be able to establish false updates of the location, thereby preventing some packets from reaching their intended destination, diverting some traffic to the intruder, or flooding third parties with unwanted traffic. Therefore, robust authentication protocols are very much needed to support the security in IP-WSN in both scenario communication between the devices and mobility, i.e., when the devices are roaming to different domain.

- Another challenge is the security privacy and trust in the IoT which can be the platform of choice for launching a variety of attacks targeting the IoT. At the most basic level, the IP-WSN devices will likely to have evolving naming and addressing schemes which can ensure that the names and addresses used are verifiable and authenticated. The research study is based on the LISP network architecture, as each LISP-capable router needs to register with a Map Server (MS), and this is known as the initial or the Registration stage. However, this stage is vulnerable to masquerading and content poisoning attacks which disclose the privacy and the trust in the WSN/IoT devices on the network. Consequently, an enhanced security protocol is needed to authenticate the MS with the source of the data (device) in a secure way. The MS and a globally distributed database that contains all known Endpoint Identifiers (EIDs) prefixes to Routing Locators (RLOC) mappings. Similar to the current Domain Name System (DNS), the Mapping systems are queried by LISP-capable devices for EID-to-RLOC mapping [Cisco-B, 2013].
- One more challenge is the data confidentiality and encryption. The WSN/IoT devices transfer the data over the network, which in turn disclose the secret information from unauthorised device. As the research study is based on LISP network architecture, the address resolving stage is therefore one of the important stages that allow the Map Server (MS) to accept Map-Requests from routers MS that has to look up the databased before returning the requested mapping to routers. However, this stage lacks data confidentiality and mutual authentication. Therefore, the implementation of

security protocol is needed to maintain the data confidentiality over the network. Besides, security protocol must be devised and applied to ensure the secure transfer of the transmitted data and guard it against unauthorized interference or misuse of the data being transmitted across the network.

1.4 Research Questions

Targeting the unaddressed challenges in the IoT Security communications using LISP, this research intended to find out answers to a set of important questions as shown below:

How to introduce an efficient architecture in term of routing size for the future of Internet and what are the main operational entities that are required in this architecture?

This research question requires the presence of a new architecture for the future of Internet in order to contain huge numbers of the IoT devices.

'What are the Security vulnerabilities/threats in terms of Internet of Things that the devices can expose in Locator/ID Separation Protocol?'

This research question requires using a specific mechanism in order to expose the security vulnerabilities for the IoT using Locator/ID Separation Protocol networks.

'How to provide End-to-End Secure communication between the Internet of Things devices?'

The answer to this question is designing a new security protocol which can provides, a high security communication to the IoT.

'Considering the proposed security protocols; how could the security protocols interface to a single refinement protocol which is integrated in order to approach a robust security for the Internet of Things using Locator/ID Separation Protocol?'

Therefore, the proposed approach aims at protecting the data and network from malicious attacks; this protection can be achieved by building a security interface between the proposed security protocols.

1.5 Research Aims and Objectives:

The aim of this research is to improve the security of the Internet of Things (IoT) networks using the Locator/ID Separation Protocol (LISP) and to provide comprehensive solution to mitigate outlined challenges. The above mentioned objectives encompass the following challenges, which must be specifically addressed:

- The LISP architecture and security protocol are still at an early stage of development and implementation as mentioned earlier. Based on this fact, this research investigated the security issues arising from deploying the LISP architecture in the IoT. The investigation comes to discover a number of vulnerabilities that must be considered before moving to the implementation stage.

- To secure the IoT and provide robust security against any security vulnerabilities, the research designed three security protocols for IoT using LISP network architecture. Firstly, security enhancement protocol is the registration protocol. It allowed the authentication of new IoT device that can join the network. This protocol was achieved by ID/Based Cryptography (IBC) [Tan et al., 2016]. Secondly, enhancement protocol attempted to secure the resolving addresses, when the devices send a data through the network, and this data must be encrypted and routers must trust and authenticate each other. This protocol was accomplished by Authentication and Key Agreement (AKA) [Li et al., 2017]. Thirdly, security protocol was developed to secure the communication between IoT devices by providing secure link and authentication. The protocol was achieved by El-Gamal cryptography [Mikhail et al 2014].

- The research proposed a set of interfaces between each level of the protocols in order to achieve a new security refinement protocol that can provide End-to-End secure communication for IoT using the LISP network architecture.

- The study then evaluated the performance of those security protocols for the IoTs. In order to resolve the problem outlined in this study, the developed protocols and their accompanying security refinement protocol were validated in terms of both (formal security analysis and AVISPA simulation tool). No security flaws were found, the performance analysis has been accompanied by Contiki and Cooja simulation tool.

1.6 Research Methodology

The methodology of this study was proposed after a detailed analysis of the identified challenges with enough consideration given to multiple parameters in each studied area. The literature review has discussed many previous related works to the present study in order to evaluate the existing limitations and eventually present suitable solutions to minimise these limitations with the intention of improving the system security.

Thus, the methodology of this research is divided into four phases. The understanding and conclusion of each phase have given motivation to address the next phase in a better manner.

- The first phase illustrates the related literature provides a solid background of the IoT in order to understand the features of this technology. Furthermore, it discussed the existing IoT network architecture and the performance limitations of this technology. Added to this, it discusses different communication security protocols for the IoT devices. A comparative analysis of these protocols and techniques has been presented in order to avoid any security vulnerabilities in the design phase.
- The second phase provides the IoT threats model which is based on LISP architecture using X.805 framework.
- The third phase of this research introduces the designed security protocols that are based on LISP architecture. These designed protocols include three security protocols; two of them are enhanced security protocols. The first protocol is to secure the Registration stage and the second is to secure the Resolving stage in LISP architecture. While the third protocol is a new security protocol that is designed to secure the communication process between the IoT devices.
- In the fourth phase, these protocols have been implemented through a simulation tool in order to achieve powerful IoT security refinement based on LISP architecture.

The comparative simulation in this thesis is performed by using the Automated Validation of Internet Security Protocols (AVISPA) tool [AVISPA, 2013], which is a formal method based tool. The AVISPA has been chosen from a range of options such as Casper/FDR, SN2 [Hossain, 2009], BAN logic [Burrows et al. 1990], and OPNET [Aboelela,2007] tools, because AVISPA provides a modular and expresses a formal language for specifying protocols and their security properties, and also integrates different back-ends that implement a variety of state-of-the-art automatic analysis techniques. Experimental results, carried out on a large library of Internet security protocols, indicate that the AVISPA tool is the state of the art for automatic security protocols. No other tool combines the same scope and robustness with such performance and scalability.

1.7 Research Contributions

In order to reach the planned objective, the following contributions are realised:

- Provided an overview of existing network architectures for the IoT and identifying the security issues and network operation in these architectures in chapter 2.

- Provided a comprehensive security analysis via using X.805 security framework to analyse the security performance of the IoT based on the LISP network architecture. Furthermore, modelled the security threats for the IoT based on LISP network architecture using AVISPA tool in order to design efficient protocols against these threats in chapter 3

- Designed security protocols for the IoT based on the LISP architecture. These protocols are divided as the following:

  1. The Initial stage; each LISP-capable router needs to be registered with a Map Server, known as the Registration stage. This stage is vulnerable to masquerading and content poisoning attacks. Consequently, the research has introduced an enhanced security protocol which addresses the security issue in the Registration stage in chapter 4.

  2. The Resolving addresses between the Routing Locators (RLoC) routers need to be addressed, e.g. when the RloC router (A) wants to send data to the RLoC router (B), both of these routers need to be authenticated so that the information can be reached from its original destination. Adding to this, LISP limits the efficiency of the security protocol which works against the redirection of the

data or acting as fake routers. Therefore, the research provides an enhanced security protocol to address this issue in chapter 5.

3. Designeing an end to end secure communication protocol for the IoT nodes. In this research study, a new security protocol provides a message authentication scheme that relies on locally shared keys and symmetric cryptographic operations only it provides also also a level of security approximating and that of end-to-end security mechanisms in chapter 6.

- The proposed protocols set an interface between each level of the protocol in order to achieve security refinement protocol to the IoT based on LISP architecture; these proposed protocols methods meet practicability, simplicity and the strong notions of security. Besides, a performance security refinement protocol analysis is provided, in order evaluated the developed protocol impact on the IP-Based Sensor Network (IP-WSN) in chapter 7.

- Provided a comparison of the recent security protocol for IoT against the present research study protocols in chapter 8.

- The proposed protocols were verified using methods based on AVISPA tool [AVISPA, 2013]. The performance and evaluation have used Contiki and Cooja simulation tool [Contiki, 2014]. Contiki and Cooja have been chosen because they are an open source operating system for the IoT. They connect tiny low-cost and low-power microcontrollers to the Internet. Indeed, they are a powerful toolbox used for building complex wireless systems.

1.8 Thesis Outline

This thesis has eight chapters, beginning with an introductory chapter one that outlines the work in the IoT technologies. It spotlights the thesis contributions, aims, objectives and motivations. Each chapter opens with an introduction and issues discussion related to the studied research area; it goes on to review the achieved security protocols in IoT after proposing suitable solutions to the raised issues. A summary is provided at the end of the chapter.

Chapter Two provides a background of the IoT that is considered the base of this research. It foucs on the previous work in this area that dealt with the existing IoT networks architecture and the performance limitations of this technology. Besides, it discusses different communication security protocols for the IoT devices. A comparative analysis of these protocols and techniques has been presented in order to avoid any security vulnerabilities in the design phase.

Chapter Three deals with the security Threats/Attacks analysis using X.805 framework for the IoT based on the LISP network architecture.

Chapter Four introduces the enhanced security protocol for Registration stage in LISP Architecture. In this chapter, more than one protocol version is provided to reveal any possible attacks on the protocol. This protocol has been designed and verified via AVISPA tool.

Chapter Five presents the enhanced security protocol for Resolving Addresses in LISP Architecture. Also, the chapter discloses different types of attacks by designing more than one protocol versions in order to test the protocol flexibility and robustness. The protocol has been verified by AVISPA tool.

Chapter Six proposes a mutual authentication protocol for IoT communication based on the LISP network architecture. In addition, the chapter presents two versions of protocol and attack revealed by AVISPA tool. The second version therefore is designed to stop any security vulnerabilities that can affect the performance of the protocol.

Chapter Seven proposes a new security refinement for the IoT based on LISP network architecture. This chapter provides a set interface of each protocol to create refinement, in order to provide a robust security to IoT devices which can resist any possible attacks by providing End-to-End Security. Also, it provides a performance analysis for the developed security refinement protocol to IP-WSN devices.

Chapter Eight provides a comparison of the recent security protocol for IoT against the present research study protocols.

Chapter Nine concludes the thesis with a summary of the main contributions of this research study along with a discussion of future works and recommendations.

Chapter 2

Internet of Things: Security Research in Heterogeneous Network

2.1 Introduction

The Internet of Things (IoT) defines a highly interconnected network of heterogeneous devices where all communications seem to be possible, even unauthorized ones. As a result, the security requirement for such network is critical whilst common standard Internet security protocols are recognised as unusable in this type of networks, particularly due to some classes of the IoT devices with constrained resources. On the other hand, for the large numbers of applications of smart object, i.e., IoT, the networking technology must be scalable, interoperable, stable, manageable and flexible. Consequently, new mechanisms are needed to protect users/devices, servers and networks infrastructure. This means that the future of the IoT networks has to integrate communication, mobility and security. Therefore, the structure of this chapter is divided as the following: 2.2 is a brief background for the IoT. In section 2.3, the IPv6 on Sensor Nodes has been tackled together with 6LoWPAN. Section 2.4, however, discusses the main security issues in the IoT. Section 2.5 provides a comprehensive study of the most common communication and mobility networks which have been used for the IoT devices and the key issues in communication and mobility. Section 2.6 discusses the type of mobility network in the IoT. Section 2.7 discusses the mobile IPv6 in the IoT and other related issues. Section 2.8 discusses the proxy MIPv6. Section 2.9 discusses the network mobility (NEMO). In section 2.10, LISP architecture is explained. Section 2.11 discusses the security in the IoT. Section 2.12 discusses the main security classification in the IoT. As for section 2.13, it verifies Security Protocol Tools. Finally, a summary concludes the chapter in section 2.14.

2.2 Brief Background

Both Machine-to-Machine (M2M) communication and IoT are results of the technological progress over the last decades, including not just the decreasing costs of semiconductor components, but also the spectacular uptake of the Internet protocol (IP) and the broad adoption of the Internet.

The application opportunities for such solutions are limited only by human imaginations. This is because; the role that M2M and IoT will have in industry and border society is just starting to emerge for a series of interacting and interlined reasons [Yang et al., 2016].

The Internet has undoubtedly had a profound impact on society and industries over the past two decades. Starting off as ARPANET [DARPA, 2017] connecting remote computers together, the introduction of the TCP/IP protocol suite, and later the introduction of services link email and the World wide Web (WWW), created a tremendous growth of usage and traffic. In conjunction with innovations that dramatically reduced the cost of semiconductor technologies and the subsequent extension of the Internet at a reasonable cost via mobile networks, billions of people and businesses are now connected to the Internet. Quite simply, no industry and no part of society have remained untouched by this technical revolution.

At the same time the Internet has been evolving, another technology revolution has been unfolding the use sensors, electronic tags and actuators to identify digitally observe and control objects in the physical world.

Decreasing rapidly costs of sensors and actuators means that where such components cost previously several Euros each, they are now a few cents only. In addition, these devices, though increasing in the computational capacity of the associated chipsets, are quite able now to communicate via fixed and mobile networks. As a result, they are able to communicate information about the physical world in near-time across networks with high bandwidth low relative cost.

So undoubtedly, M2M solution will be seen for quite some time; we are now entering a period of time where the uptake of both M2M and IoT solutions will increase dramatically. The reasons for this are diagnosed as the following [Singh, 2012]:

- An increased need for understanding the physical environment in its various forms, from industrial installations through to public spaces and consumer demands. These requirements are often driven by efficiency improvements and sustainability objectives, or improved health and safety.
- The improvement of technology and improved networking capabilities.

- Reduced costs of components and the ability to collect and analyse the data they produce more cheaply.

2.2.1 Machine-to-Machine

M2M refers to those solutions that allow communication between devices of the same type and specific application, all via wired or wireless commutation network. M2M solutions allow end to capture data about events from assets such as temperature or inventory levels. Typically, M2M is deployed to achieve productivity gains, reduce costs and increase safety or security. In fact, M2M has been applied in many different scenarios, including the remote monitoring and control of enterprise assets, or to provide connectivity of remote machine-type devices [Atanasoy et al., 2017]. Remote monitoring and control have generally provided the incentive for industrial applications, whereas connectivity has been the focus in other enterprise scenarios such as connected vending machines or point-of-sale terminals for online credit card transactions. M2M solutions, however, do not generally allow for the broad sharing of data or connection of the devices in question directly to the Internet [Yang et al., 2016].

2.2.2 Internet of Things

The IoT is a widely used term for a set of technologies, systems and design principles associated with the emerging wave of Internet-connected things that are based on the physical environment. In many respects, it can initially look the same as M2M communication connecting sensors and other devices to Information and Communication Technology (ICT) systems via wired or wireless networks [Zigler et al., 2015].

In contrast to M2M, however, IoT also refers to the connection of such systems and sensors to the broader Internet, as well as the use of general Internet technologies. In the longer term, it is envisaged that IoT ecosystem will emerge not different to today's Internet, allowing things and real world objects to connect, communicate and interact with one another in the same way humans do via the web today [Clark et al., 2003]. Increased understanding of the complexity of the system in question, economies of scale and methods for ensuring interoperability in conjunction with key business drivers and governance structures across value chains will create wide-scale adoption and deployment of IoT solutions [Mahalle et al, 2014]. The Internet will no longer be only about people, media and content, but it will also include all real-world assets as intelligent creatures exchanging information, interacting with people, supporting business processes of enterprises and creating knowledge as shown in figure 2.1. The IoT is an extension to the existing Internet [Makinen, 2014].

IoT is actually about the technology; the remote monitoring and control is also about where these technologies are applied. IoT can have a focus on the open innovative promises of the technologies at play and also on advanced and complex processing inside very confined and close environments such as industrial automation. Visions put forward have included notions like a global open fabric of sensor and actuator services. These notions integrate many Wireless Sensor Network (WSN) deployments and provide different levels of aggregated sensor and actuator series in an open manner. The purpose is to achieve application innovation and use it not only in pure monitor and control type of applications, but also to enrich other types of services with contextual information [Kortuem et al., 2010]. However, IoT applications will not only rely on data and services from sensor and actuators alone. Equally important is the blend in of other information sources that have relevance from the viewpoint of the physical world. These can be data from Geographic Information Systems (GIS) like road databases and weather forecasting systems and can be of both a static nature and a real-time nature. Even information extracted from social media like Twitter feeds or Facebook status updates that relate to real world observations can be fed into the same IoT system [Butun, 2017].

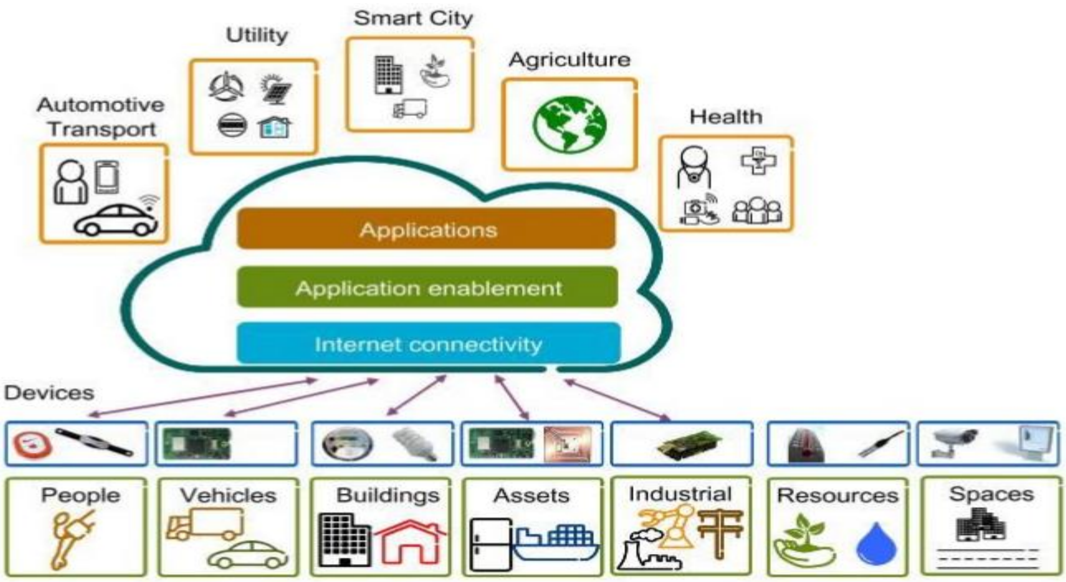


Fig 2.1 Internet of Things [Kogatam, 2014]

2.3 The IPv6 on Sensor Nodes: 6LoWPAN

Internet Protocol version 6 (IPv6) is the most recent version of the Internet Protocol (IP). In other words, it is the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet [Nicills et al., 2017]. With the introduction of 6LoWPAN compressed IPv6 in WSNs, resource constrained devices can be connected to the Internet [Khan et al., 2017]. This hybrid network of the Internet and the IPv6 connected constrained devices from the IoT [Sanchez, 2015]. Unlike the Internet where devices are mostly powerful and unlike typical WSN where devices are mostly resource constrained, the things in the IoT are extremely heterogeneous [Dawood et al., 2014]. The IoT device can be a typical sensor node, a light bulb, a microwave oven, an electricity meter, an automobile part, a smartphone, a PC or a laptop, a powerful server machine or even a cloud [Rantos et al., 2014]. Hence the numbers of potential devices that can be connected to the IoT are in hundreds of billions. Definitely, this requires the use of IPv6 [Xiaorong et al., 2013], a new version of the Internet Protocol that increases the address size from 32 bits to 128 bits (2^{128} unique addresses). Also, a number of protocols are being standardized to fulfil the specific needs of the IoT [Dooley et al., 2013].

6LoWPAN allows the transmission of IPv6 packets over an IEEE 802.15.4 network. The main idea of 6LoWPAN is to introduce an adaptation layer to enable IPv6 communication in WSNs.

2.4 Security Issues in Internet of Things

The security in the IoT is essentially linked to the ability of users to trust their environment. If users do not believe their connected devices and their information are reasonably secure from misuse or harm, the resulting erosion of trust causes a reluctance to use the Internet [Pishya 2017]. This has global consequences to electronic commerce, technical innovation, free speech and practically every other aspect of online activities. Indeed, ensuring security in the IoT products and services should be considered a top priority for the sector [Cheah, 2017].

However, as we increasingly connect devices to the Internet, new opportunities to exploit potential security vulnerabilities grow. Poorly secured IoT devices could serve as entry points for cyberattack by allowing malicious individuals to re-program a device or cause it to malfunction. It is noteworthy that the poorly designed devices can expose user data to theft by leaving data streams inadequately protected. Added to this, Failing or malfunctioning devices can also create security vulnerabilities. These problems are just as large or even larger for the small, cheap and ubiquitous smart devices in the IoT as they are meant for the computers that have been traditionally the endpoints of Internet connectivity. Along with potential security design deficiencies, the sheer increase in the number and nature of the IoT devices could increase the opportunities of attack. When coupled with the highly interconnected nature of the IoT devices, every poory secured device that is connected online will affect potentially the security and resilience of the Internet globally, and not just locally. Therefore, two main security issues can be summarised as the following:

- **Security Issue 1:** Many IoT devices such as sensors and consumer items are designed to be deployed at a massive scale which is beyond that of traditional Internet-connected devices. As a result, the potential quantity of interconnected links between these devices is unprecedented. Furthermore, many of these devices will be able to establish links and communicate with other devices on their own in an unpredictable and dynamic fashion. Consequently, security communication protocol is mostly required in the IoT. It is important that a receiver which is able to verify the IoT data is generated by trusted nodes. It is also necessary to encrypt the IoT data in transit. As this research study focuses on the network layer of the Open Systems Interconnection Model (OSI Model), the IPsec works well on non-Low-power devices which are not subject to severe constraints on host software size, processing and transmission capacities. IPsec supports Authentication Header (AH) for authenticating the IP header and Encapsulating Security Payload (ESP) for authenticating and encrypting the payload [Djeddai et al., 2016]. The main issues of IPsec are twofold processing power and key management. Since these tiny IoT devices do not process huge number of data or communicate with many different nodes, it is not well understood if complete implementation of Security Association Database (SADB), policy-debase and dynamic key-management protocol are suitable for these small battery powered devices. In addition, given existing constraints in IoT environments, IPsec might not be suitable to be use in such environments, especially that IoT node devices might be able to operate all IPsec algorithms on its own capability either Full-Function Device (FFD) or Reduced-Function Devices (RFD). Bandwidth is a very rare resource in IoT environments. The fact that IPsec requires another header (AH or ESP) in every packet makes its use problematic in IoT environments [Aouini et al., 2016]. Besides, IPsec requires two communicating peers to share the secret key that is established dynamically with the Internet key Exchange (IKEv2) protocol. Thus, it has an additional packet overhead incurred by IKEv2 packets exchange [Rao et al 2016].

- **Security Issue 2:** In the IoT, there are privacy, integrity and disclosed data (confidentiality) issues. This is because of the features of the IoT as they are randomly deployed by users in the network. Thus, the decentralisation of these nodes in most cases can create many privacy and security issues. Therefore, the majority of the IoT applications need to take into considerations the support of mechanisms to carry out the authentication, authorization and key management. In addition, due to the reduced capabilities from the constrained devices enabled with Internet connectivity, a higher protection of the edge networks needs to be considered with respect to the global network. In other words, during the development of IPv4, information Technology security is not one of the focus points [Siddika et al., 2017]. This had caused a lot of vulnerabilities that can be exploited in IPv6 implementation. Even though a lot of methods such as Secure Sockets Layer (SSL) and Secure Shell (SSH) are introduce to overcome this weakness, still it they are not sufficent. In a network, five phases are involved in exiting IPv4 environment [Shaharuddin et al., 2017], which are reconnaissance, scanning, Gaining access Maintaining access, and Clearing track. The first two phases involve a process of scanning for vulnerabilities in the host networks. One of these vulnerabilities are exposing the process of gaining access is executed. Freely available on the internet, port scanning tools like Nmap [Jicha et al., 2016] and Wireshark [Beeharry et al., 2016] can bused to execute the reconnaissance and scanning phase. As the number of IPv4's addresses is small, scanning a class C network takes only a few minutes. This show how vulnerable is IPv4 network, with a few minute all open access can be expose. Furthermore, Address Resolution Protocol (ARP) and internet Control Message Protocol (ICMP) are two protocols in layer 4 used for finding a host's hardware address and protocol for responding to errors in datagram respectively [Nikolenko et al., 2016]. In an attack these two protocols can be exploited by associating the packet with a fake address resulting in other packet to be mistakenly sent to a rogue address. Besides, IP fragmentation referred to IP datagram that is broken up to smaller size. This is to enable the IP datagram to pass through the data link medium which has a limit on the size of transiting frame called

Maximum Transmission unit (MTU). An attacker can use this features to evade from being detected by firewall and Network Intrusion Detection System (NIDS). Also, broadcast features in IPv4 can also be exploited. A huge number of frames are broadcasted through a network and are now flooding the network system. However this large number of frames looks like a legitimate packet will hinder the hosts from receiving a valid packet. Likewise, the packet in IPv4 can be intercepted during its transmission by eavesdropping on the network. This attack is called Man in the Middle Attack (MitM) and it occurs as a result of the lack of authentication mechanism provided in IPv4 protocol [Shuai et al., 2016]. And it can be done applying ARP attack as explained above.

In order to prevent all vulnerabilities, extra tools are used to harden the network security in IPv4 environment. Application like Network Address Translation (NAT) is used to overcome the shortage of IPv4's addresses. Firewall and IDS are deploying to protect and detect any anomalies in the networks. Access Control List (ACL) can be applied in network to drop any packets that can cause security problems in the networks.

2.5 Communications and Mobility in the Internet of Things

A Wireless Sensor Network (WSN) is a multi-hope network and is made up of a number of wireless sensor devices. With the constant increase in demands for various global services, it has become urgent to connect these wireless sensor devices to Internet [Khan et al., 2014]. Since IP is the facto standard for the Internet, it is quite reasonable to believe that IP could be the future for WSN [Bag et al., 2009]. However, and due to limited resources, implementing the heavy IP protocol in WSNs has become a big challenge [Pradeska et al., 2017]. The IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) enables the wireless connection to be one of the key technologies for WSNs [Twayej et al., 2017]. Therefore, this section, reviews the recent mobility and communication architecture that supports the IoT devices and defines the suitable network architecture to the IoT network.

2.6 Types of Mobility in Internet of Things

The two types of mobility that are possible in 6LoWPAN networks itself are micro and macro mobility [Shelby et al., 2009]. Micro mobility in 6LoWPAN refers to the mobility of a node within 6LoWPAN where IPv6 prefix remains the same. Likewise, macro mobility refers to the mobility between two 6LoWPAN networks with different IPv6 prefix. In this case, the handover is presented, while in the second, joint roaming and handover mobility are presented in place as figure 2.2 shows. The same definition regarding macro and micro mobility of nodes can be applied for the edge routers. From the network perspective, there is both node and network mobility. Network mobility occurs when the edge router changes its point of attachment [Mulligan et al., 2010], while all nodes from 6LoWPAN network remain essentially the same.

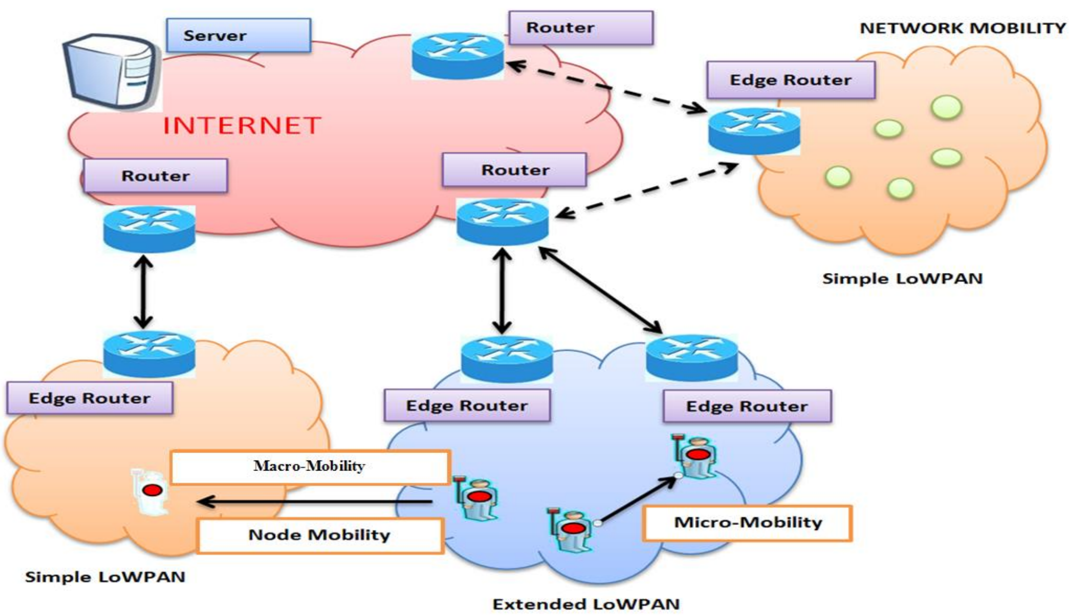


Fig 2.2 Mobility Types Inside 6LoWPAN [Shelby et al., 2009]

The above Figure 2.2 shows the two types of mobility, i.e., the macro and the micro. In case of macro, when the edge router changes its point of attachment, the IPv6 address also changes and this result in a change of node's IP addressing. In other words, once the node changes its point of attachment, there are several things to be done in order to resume data flows: re-establishing a link by assigning IPv6 address by bootstrapping node, updating of DNS (Domain Name System) settings with new IPv6 and notification to application layers etc. When micro mobility takes place, link layer is sufficient to cope with mobility without any notification to the network layer. In 6LoWPAN networks, some techniques can be used in future to deal with micro mobility inside LoWPAN itself.

IEEE 802.15.4, a standard which specifies the physical layer and media access control for low-rate wireless personal area network) intends to leave mobility issues to the network layer, and all topology changes are node controlled. Neighbour Discovery (ND) [Mulligan et al., 2010] for 6LoWPAN is used to cope with a micro mobility in extended LoWPAN networks. Here, ND proxy technique and synchronization are used between routers allowing a node to save the same IPv6 address, no matter where the point of attachment is. In contrast to this, macro mobility always includes changes of IPv6 address of a node. This is especially hard to deal with from the perspective of an application. If a node is acting as a client, the best way that also fits 6LoWPAN is that whenever node detects change in IPv6 address, the application restarts itself. However, this is not so practical for the case where the node is acting as a server due to needs that servers must be reliable 100% of time. In this scenario and within 6LoWPAN network, application is dealt on application level using Session Initiation Protocol (SIP) which is a communications protocol for signalling designed, for the purpose of controlling multimedia communication sessions. SIP can support any type of single-media or multi-media session, including 6LoWPAN [Zhang et al., 2015], Uniform Resource Identifier (URI). The latter is a string of characters used to identify a resource. Such identification enables interaction with representation of the resource over a network, typically the World Wide Web, using specific protocols. Therefore, URI is important to 6LoWPAN to as it access the resource of the devices online [Liu et al., 2016]. Furthermore, when the user enters the URL in the Web browser, the Domain Name System (DNS) server uses its resources to resolve the name into the IP address for the appropriate Web server. The user can get just connected through a domain name server, also called a DNS server or name server, which manages a massive database that maps domain names to IP addresses [Fireze et al., 2011].

2.7 Mobile IPv6

Mobile IPv6 basic operations consist of the following steps. First, the mobile node should detect the arrival in a new IPv6 network. This step is known as movement detection and it is based on the reception of a new router advertisement. Then, the mobile node can create a new care-of address and perform a duplicated address detection procedure to ensure that this address is unique on the link. Finally, the mobile node registers this new care-of address to the home agent by the means of binding update and binding acknowledgment messages. Mobile IPv6 also defines an alternative to bidirectional tunnel known as route optimization [Imran et al 2016]. Corresponded node is a node that is intended to communicate with mobile node it may be mobile or a stationary node. In this mode, the mobile node and its correspondent communicate directly without the help of the home agent. For this, the mobile node registers its current binding at the correspondent by exchanging binding update and acknowledgment. Once the registration with a correspondent is complete, the mobile node sends its data to this correspondent by using its care-of address as source address and adding its home address in a destination option extension header. Similarly, the correspondent sets the care-of address of the mobile node as the destination address and adds the home address in a routing type 2 header. The route optimization mode allows the shortest communications path to be used between the mobile node and its correspondent and reduces the congestion potentially experienced by the home agent. The above discussed processes are all presented in the following Figure 2.3.

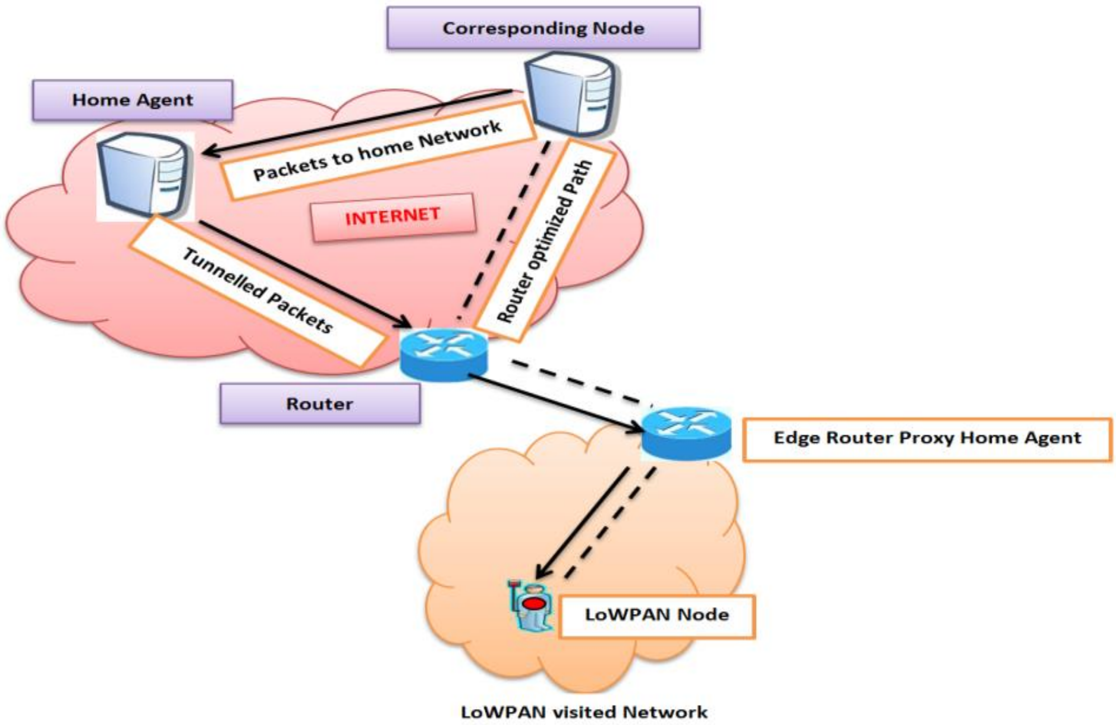


Fig 2.3 MIPv6 in 6LoWPAN [Shelby et al., 2009]

However, each correspondent is required to implement additional mechanisms to support the route optimization of Mobile IPv6. At each handover, the mobile node is also required to update its bindings at all of its correspondent which may generate large control traffic overhead. Due to those limitations, any network protocol should protect itself against misuses of its features and mechanisms. In Mobile IPv6, binding updates with the home agent should use an IPsec Encapsulating Security Payload (ESP) security association to protect the integrity and authenticity of the binding updates and acknowledgments [Raza et al., 2011]. Similarly, the binding updates with correspondent are secured by the return rout ability procedure which ensures that the sender of a new binding update is reachable through both its home address and claimed care-of address. Security consideration in 6LoWPAN is currently investigated by the research community [Imran et al., 2016]. However, when IPsec ESP is used, the following headers, e.g. mobility header, inner IPv6 header, transport header, etc. are encrypted and as such cannot be compressed by 6LoWPAN [Isah et al., 2015]. Resulting packets would only leave few bytes for data and the overall system is likely to generate fragmentation. Security consideration in Mobile IPv6 for the IoT and 6LoWPAN is therefore really challenging and it requires new architecture network that is integrated with the IoT.

2.8 Proxy MIPv6

In M2M world, it is quite often to change a point of attachment in the same domain, (for example the operator), as IETF has devolved standardized Proxy MIPv6 that consists of a local hierarchical structure of routers which handles mobility on behalf of nodes. It is very suitable, therefore, to be used for LoWPAN networks as it allows LoWPAN edge routers to proxy MIPv6 for attached LoWPAN nodes. The PMIPv6 allows LoWPAN to communicate with different networks request some services, such as downloading file, making connection, web page, or other resources from different servers [Huang et al., 2017].

Figure 2.4 shows the mobility aspect of the LoWPAN devices in PMIPv6. The transaction messages have been summarised in 4 steps as the following: Step (1) assumes that the LoWPAN node wants to change its position to a new segment on a different subnet. Step (2) PMIPv6 uses Router Solicitation/Router Advertisement (RS/RA) communication between mobile nodes and Mobile Access GWs (MAGs) to detect when one of the mobile nodes has changed its point of attachment. In order to apply MIPv6, each LoWPAN router must act as MAG providing separate 64 bit prefix address for each mobile node. Step (3) Mobile Access GWs send proxy binding updated towards Local Mobility Anchor (LMA) on behalf of the mobile nodes attached to them. Step (4) the mapping is then done in the LMA between this address and the temporary address of the visited MAG. In each moment, there is a bidirectional channel between MAGs and LMA enabling LMA to send traffic towards mobile nodes static addresses (mobile node Home Address). Moreover, the basic idea is that the edge routers in 6LoWPAN, which are full of IPv6 (MIPv6 as well) sufficient to cope with network mobility in general, i.e., routers and nodes attached to them.

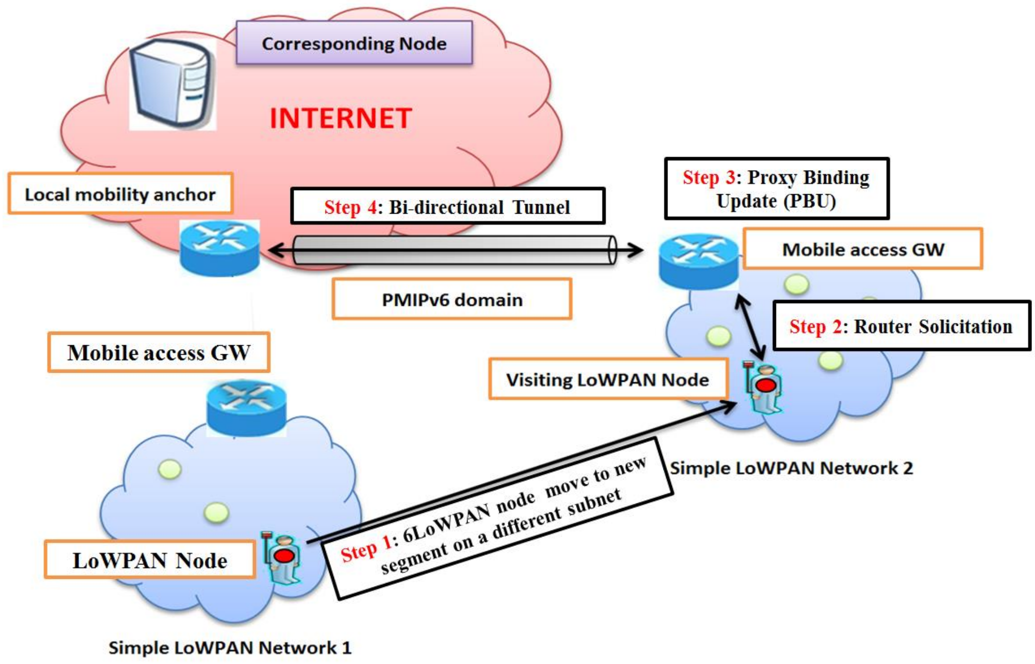


Fig 2.4 PMIP in 6LoWPAN [Shelby et al., 2009]

However, in MIPv6 the signalling messages, Proxy Binding Update and Proxy Binding Acknowledgement, exchanged between the mobile access gateway and the local mobility anchor must be protected using end-to-end security association offering integrity and data origin authentication. Therefore, as in Mobile IPv6, the use of the IPsec is to protect a mobile node's data traffic in PMIPv6 [Raza et al., 2017].

2.9 Network Mobility (NEMO)

To begin with, NEMO has introduced the term mobile router and mobile nodes within mobile network and are called Mobile Networks Nodes (MNN). If NEMO is applied in the 6LoWPAN network, even though each 6LoWPAN node is not a running mobility protocol, it can keep up session continuity for all the mobile network nodes and even when the mobile router changes dynamically its point of attachment to the Internet through the 6LoWPAN mobile router [Ye et al., 2017]. NEMO protocol [Hasan et al., 2017] enables the extension of the home agent so that the agent becomes able to work with prefixes as with Home Addresses of mobile nodes. Figure 2.5 shows the communication flows between nodes when using NEMO. The NEMO is applied when mobile router, in its communication with home agent, negotiates the prefixes which are forwarded back to it. Home agent then forwards all packets that match with bound prefix of MNNs towards mobile router. This can be a good solution for network mobility in 6LoWPAN when mobile nodes and edge router all together are changing their point of attachment. In 6LoWPAN, edge router becomes a mobile router that binds new address in the visited network with home LoWPAN prefix as shows in figure 2.5. In practice, this means that there is no change visible inside LoWPAN network when network mobility occurs, because LoWPAN still uses the same prefix as in its home network. Home agent then transfers all the data destined to the same prefix using a tunnel between HA and edge router. The disadvantage of NEMO is that it cannot deal with individual node mobility on behalf of LoWPAN nodes unless MIPv6 is installed itself on nodes or if home agent or PMIP is used. Prefix delegation can be done by The Dynamic Host Configuration Protocol version 6 (DHCPv6).

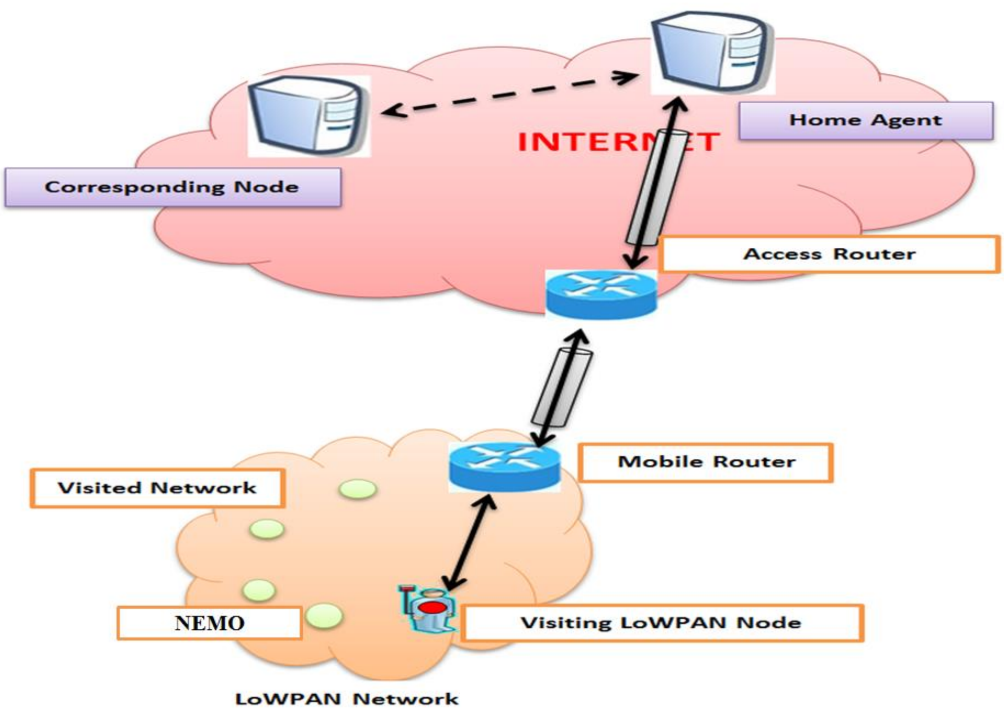


Fig 2.5 NEMO in 6LoWPAN [Shelby et al., 2009]

However, in terms of security, these binding updates are vulnerable to different attacks as many malicious users send fabricated binding to fool the Mobile Router (MR), the Home Agent (HA) and the Corresponding Node (CN). Although the path between the MR and the HA is protected by IPsec tunnel, the paths between the MR and CN, between the HA and CN and between a mobile network node and the MR remain unprotected. Nonetheless, IPsec will be very heavy and expensive to use on the IoT or 6LoWPAN network.

2.10 Locator/ID Separation Protocol

Locator/ID Separation Protocol (LISP) is another approach to split the current Internet namespace into separate identifier and addressing entities that have been proposed [Kafle et al., 2010]. In order to understand the Architecture LISP, it is necessary to note that LISP uses dynamic tunnelling encapsulation approach rather than requiring the pre-configuration of tunnel end-points. It is, in fact, designed to work in a multi-homing environment and support communications between LISP and non-LISP sites inter working [Cisco-A, 2014].

To improve routing scalability while facilitating flexible address assignments in multi-homing and mobility scenarios, the LISP describes changes to the Internet architecture in which IP addresses are replaced by Routing Locators (RLOCs) for routing through the global Internet and by Endpoint Identifiers (EIDs) for identifying network sessions between devices [Cisco-A, 2014]. As shown in Figure 2.6, three important components exist in the LISP environment: the LISP sites (EID space), the non-LISP sites (RLOC space) and the LISP Mapping System which includes Map Servers and databases [Raheem et al., 2014]. The three components are explained as the following:

- **The LISP sites (EID space):** These represent customer end-sites in exactly the same way that end-sites are defined today. However, the IP addresses in the EID space are not advertised to the non-LISP sites, but are published in the LISP Mapping Systems which perform the EID-to-RLOC mapping. The LISP functionality is deployed on the site's gateway or edge routers. Therefore, based on their roles, two types of routers are defined and as follows: Firstly, the Ingress Tunnel Routers (ITRs) which receive packets from hosts and send LISP packets towards the Map Server. Secondly, the Egress Tunnel Routers (ETRs), which receive LISP packets from the Map Server and pass them to hosts [Cisco-A, 2014]

- **Non-LISP sites (RLOC space):** They represent current sites where the IP addresses are advertised and used for routing purposes.
- **LISP Mapping Systems:** These are represented by Map Servers (MS) and a globally distributed database that contains all known EID prefixes to RLOC mappings. Similar to current DNS, the Mapping systems are queried by LISP-capable devices for EID-to RLOC mapping.

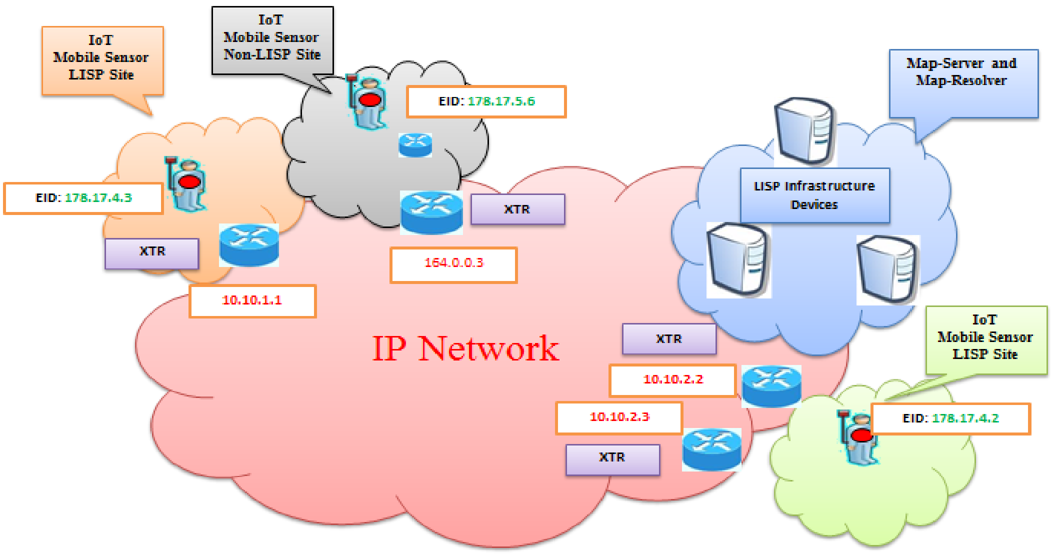


Fig 2.6 the LISP Network Architecture Design [Cisco-A, 2014]

2.10.1 Address Registration Procedure:

The functionality of the LISP goes through two stages:

1. The EID prefix Configuration and ETR Registration Stage :

As explained in [Cisco-A,2014], the EID Prefix Configuration and ETR Registration Stage is an ETR publishes its EID-prefixes on a Map Server (MS) by sending LISP Map-Register messages which include the ETR's RLOC and a list of its EID-prefixes. Initially, it has been presumed that prior to sending a Map-Register message, the ETR and the Map Server must be configured with shared secret or other relevant authentication information. Upon the receipt of a Map-Register from an ETR, the Map Server checks the validity of the Map-Register message and acknowledges it by sending a Map-Notify message. When registering with a Map-Server, an ETR might request a no-proxy reply service which implies that the Map Server will forward all the EID-to-RLOC mapping requests to the relevant ETR rather than dealing with them.

Therefore, Figure 2.7 shows the registration operation which can be summarised as the following; Step (1) the Egress Tunnel Router (ETR) detects the new host Endpoint Identifiers (EID) 178.17.4.2/24 on the network. Step (2) Routing Table (RT) updates and registers the new EID with the current Router Locator (RLOC). Step (3) The ETR sends Map-Register (MR) messages including the ETR RLOC and a list of its EID to the Map-Server (MS). Step (4) MS checks the validly of a Map-Register message and acknowledges it by sending a Map-Notify (MN) message to the ETR.

However, the security-related research is still at an early stage. The research in [Raheem et al, 2013] has highlighted the potential threats to be addressed at a later stage of this research. Therefore, the main security registration issues is whern an LISP-capable router publishes all its hosts EID to the Map Server via a Map–Resister as is shows figure 2.7 in step (3). For a secure Registration, two information elements are critical: the hosts EID and the routers address RLOC. Certainly, a malicious router might spoof different RLOC and supply wrong EID-Prefixes to the MS. This is very similar to poising attacks against Domain Name Server (DNS) or router table [Maino et al., 2012]. To stop such attacks, this research has provided a security enhancement protocol to the registration stage. It allowed the authentication of new IoT device that joins the network. This protocol was achieved by ID/Based Cryptography (IBC) [Tan et al., 2016]. The IBC helps to certify the messages sender as the real owner of the RLOC that will update the Map Server. The main advantage of using the IBC over traditional Public Key Infrastructure (PKI) is that since the public key will be derived from the nodes identifiers, IBC eliminates the need for a public key distribution infrastructure; more details are in section 2.12.1.1

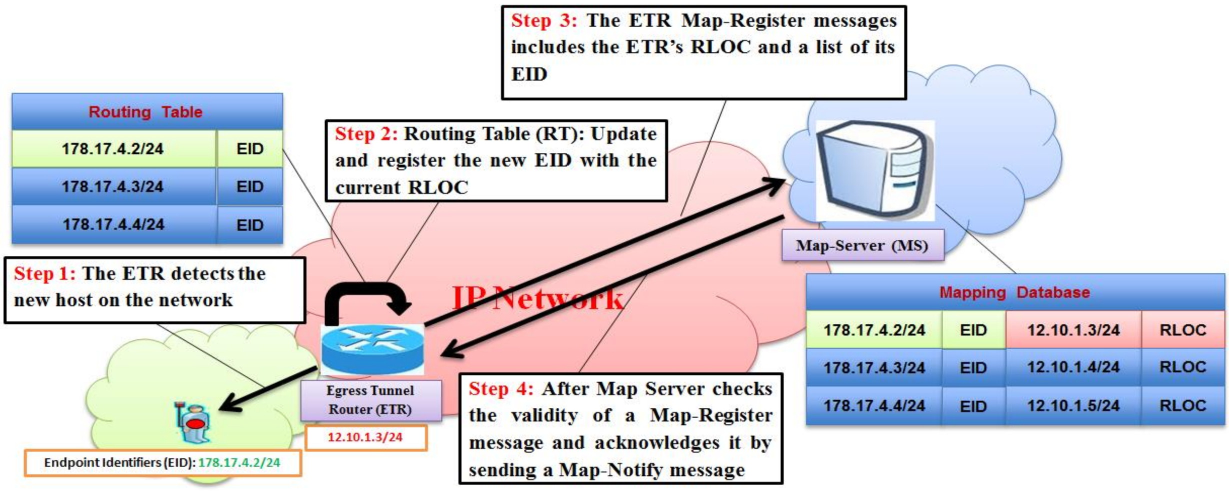


Fig 2.7 ETR Address Registration Procedure

2. The Address Resolving Stage:

Once a Map Server has EID-prefixes registered by its client ETRs, the following resolving operational steps take place, as figure 2.8 shows. Step (1) the Ingress Tunnel Router (ITR) send Map-Request to the Map-Server (MS), then the MS first checks to see if the required EID matches a configured EID-prefix. If there is no match, the Map Server returns a negative Map-Reply message to the ITR. Step (2) In case of a match, the MS re-encapsulates and forwards the resulting Encapsulated Map-Request to one of the registered ETRs. Step (3) ETR return Map-Replay directly to the requested ITR. Step (4) Data exchange between ITR and ETR [Cisco-A, 2014].

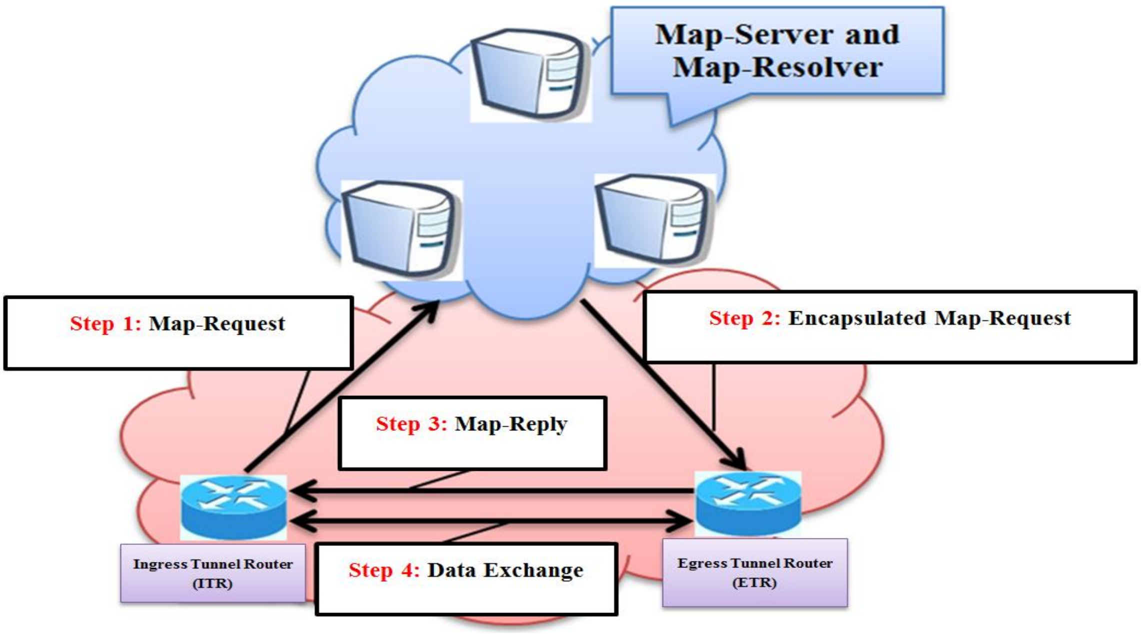


Fig 2.8 the No Proxy Map Server Processing

The main security resolving stage issues occurs as, there is no mechanism such as encryption/ or encoding between the ITR and ETR routers. And this causes a lack of data confidentiality and mutual authentication. Therefore, the attacker is able to capture and modify all the packets exchanged between an Ingress Tunnel Router (ITR) and Egress Router (ETR) and between the X Router Tunnel Router (XTR) and the mapping system (MS). Thus, an enhancement protocol attempts to secure the resolving addresses, when the devices send a data through the network; this data must be encrypted and routers must trust and authenticate each other. This protocol was accomplisheqd by Authentication and Key Agreement (AKA) [Li et al 2017].

2.10.2 Address Query and Communication:

Figure 2.9 shows the communication between Mobile Sensor devices in the case of LISP enabled sites. To explain this, it is better to assume that Mobile Sensor Node (MSN) EID 1 wants to communicate with WSN EID 2. In order to establish the communication, the following steps show the procedures of establishing this communication [Fuller et al., 2013]:

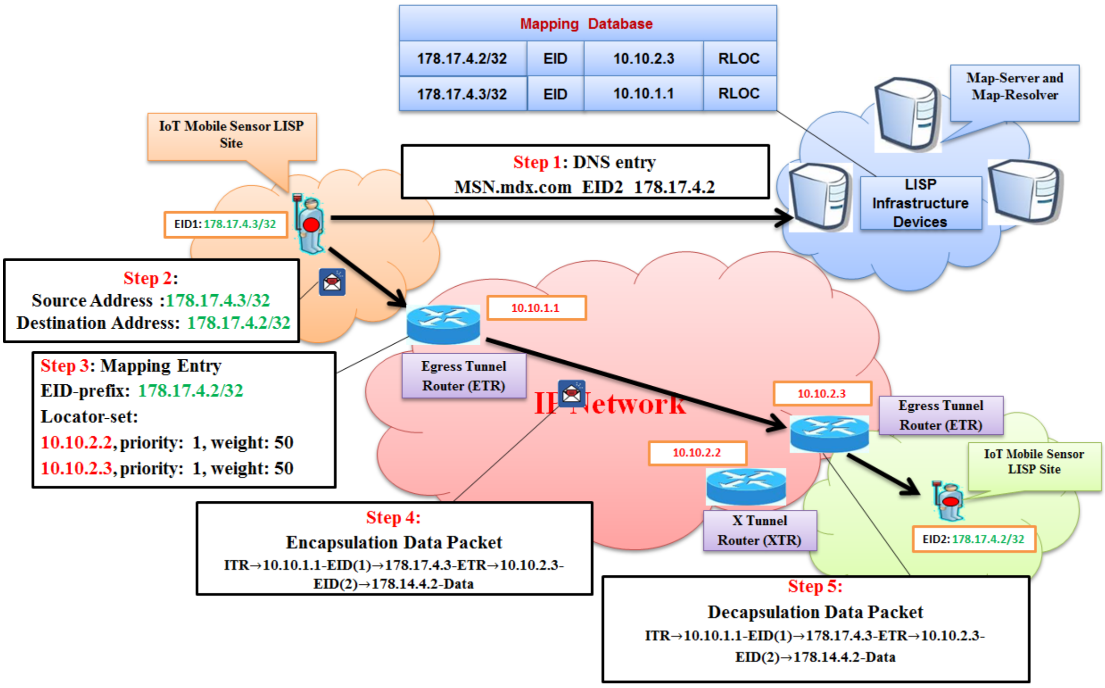


Figure 2.9 The Communication between Mobile Sensor devices (LISP Sites)

Step (1): the MSN device which is in the remote LISP enabled site sends queries through DNS to get the IP address of the destination server that is deployed at the LISP enabled EID 2. Step (2): the traffic that originated from the devices (MSN) is steered towards the Local LISP enabled device (usually devices default gateway). However, the LISP device performs first a lookup for the destination 178.17.4.2 in its routing table. Step (3): the ITR receives valid mapping information from the mapping database and populates local map-cache. If the mapping exists, the packet is encapsulated using that map-cache policy and then forwards it. If no mapping exists, the ITR sends a map request for the destination EID in query to its configured Map-Resolver. Furthermore, each entry has associated priority and weight values that are controlled by the destination site to influence the way inbound traffic is received from the transport infrastructure. The priority is used to determine if both ETR devices can be used to receive LISP encapsulated traffic destined to Local EID subnet. The weight allows tuning the amount of traffic received by each ETR in load-balancing which is shown in Figure 2.9. Step

(4): the ITR performs LISP encapsulation of the original IP traffic and sends it to transport infrastructure, destined to one RLoCs of the EID 2 ETR. Step (5): The ETR receives the packet, de-capsulate it and sends it to the site towards the destination EID2.

2.10.3 Mobility Transaction:

Figure 2.10 shows the mobility aspect of the IoT devices in LISP network architecture. The mobility procedure has been summarised in 6 steps as the following: Step (1) assumes that the Mobile Sensor Node (MSN) EID 2 wants to change its position to a new segment on a different subnet. Step (2) the 10.10.2.2 is a LISP router which is configured with a dynamic-EID range of addresses that are acceptable to move, and then the XTR detects the new host movement. Step (3) Once XTR 10.10.2.2 notices that it has a new server, it updates the new EID and it installs a specific /32 address in its routing table. Step (4) the XTR 10.1.0.2.2 sends queries to register the /32 address with the map-register message with the map server, then the map database checks the data and updates old location with the new location. Step (5) the map server now sends a map-notify message back to the 10.10.1.1 XTR and lets it know that it is no longer responsible for 178.17.4.2. Step (6) the XTR 10.10.1.1 updates the routing table and removes old EID by installing ‘null0’.

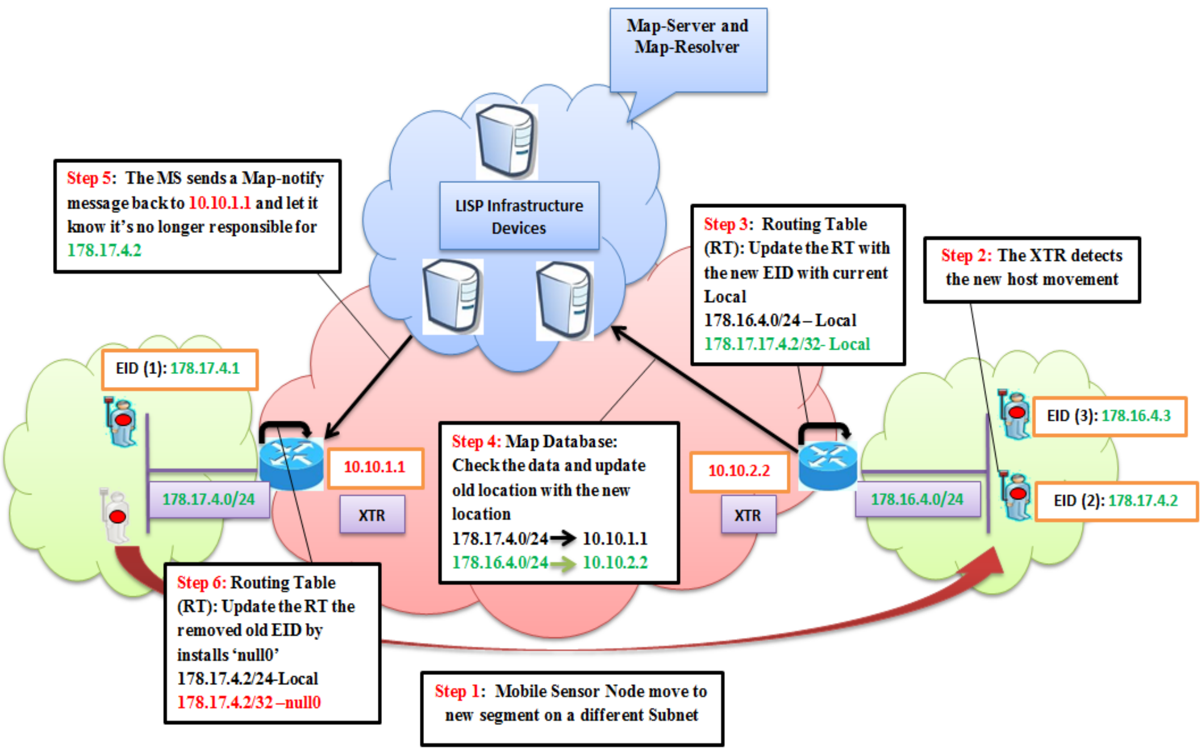


Fig 2.10 Mobility Signalling

Consequently, LISP network architecture supports IoTs/6LoWPAN devices in both communication and mobility and it adopts a huge number of devices at the same time. However, the LISP is still at an early stage of implementation and the protocols security, in particular, is still in its first beginning [Raheem et al., 2014].

2.11 Security in Internet of Things

As demonstrated earlier in the study, the IoT offers connectivity for both human-to-machine and machine-to-machine communications. Moreover, everything in the near future is likely to be equipped with small embedded devices which are able to connect to the Internet. Such ability is useful for various domains in our daily life; from building automation, smart city and surveillance system to all wearable smart devices [Roman et al., 2011]. However, the more the IoT devices are deployed, the greater our information system is at risk. Indeed, a significant number of devices in IoT are vulnerable to security attacks, such as DoS and replay attacks, security attacks that are the result of their constrained resources and the lack of protection methods. As such, they will definitely lead to sensor battery depletion and intern to poor performances of sensing application. However, the IoT security has been one of the most discussed and yet pending issues, even after the existence of protocols for IPv6 network security such as IPSec, and also for datagrams, i.e., UDP or CoAP [Betzler et al., 2016] such as DTLS [Ngoepe et al., 2017]. Security for the IoT is not excessively extended and deployed because of the difficulties in configuring (IPSec) for end users and the lack of scalable certificate management for DTLS [Kothmay et al., 2013]. Consequently, the majority of the Internet traffic continues being transmitted in plain text, i.e., unprotected. Thus, to provide the security in IoT such as End-to-End (E2E) communications, it is necessary to clarify and explain the currently proposed security protocols in this technology. Section 3.10 discusses the main security protocols classification in the IoT.

2.12 The Security Protocols Classification in Internet of Things

The existing security proposers in the IoT are categorized into two main types: security that relies on asymmetric key schemes and security that pre-distribute symmetric keys as shown in Figure 2.11. This section describes the two first levels of the security taxonomy in the IoT.

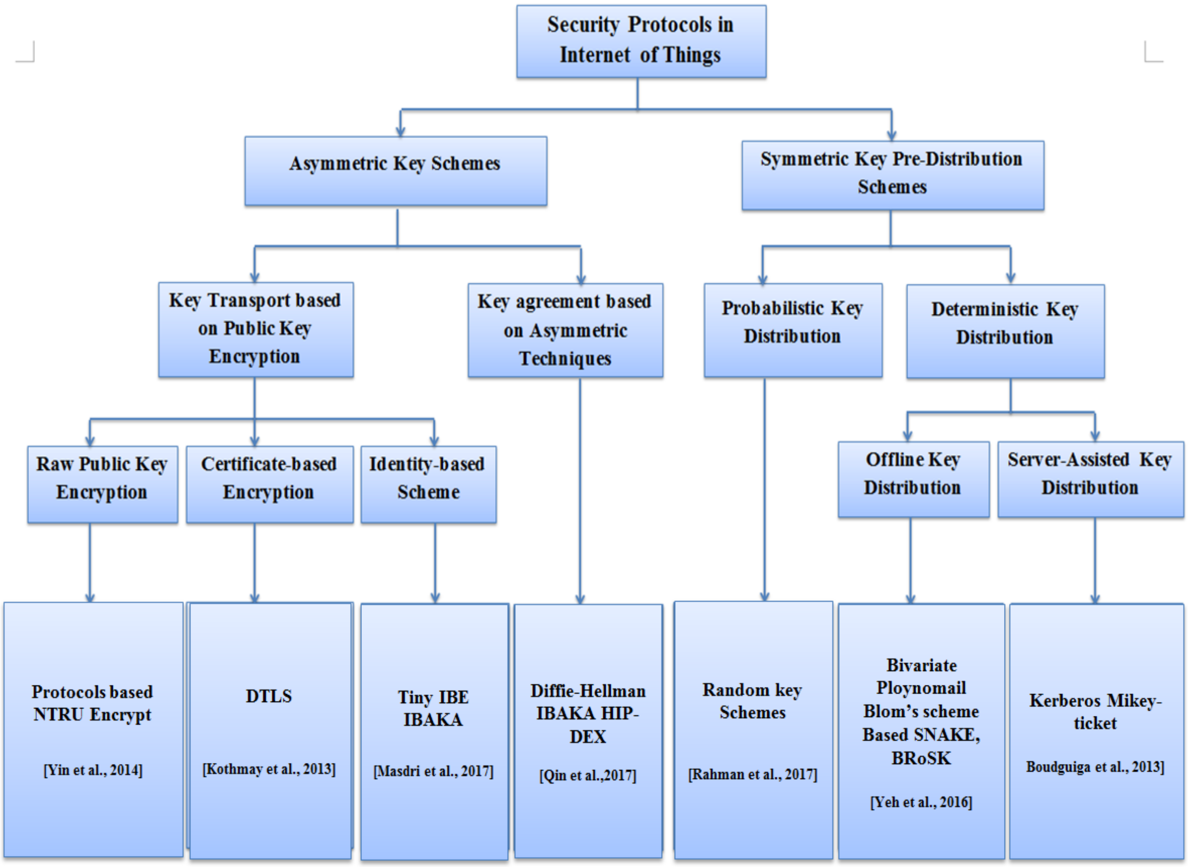


Figure 2.11 Security Classification of IoT Network

- Asymmetric Key Schemes (AKS): The key schemes based on asymmetric cryptography, also known as Public-Key Cryptography (PKC) are considered as a very common approach to establish a secure communication between two (or more) parties. They employ asymmetric algorithms and are widely deployed in the conventional Internet. The applicability AKS in IoT has one major inconvenience, which is the computation cost and energy consumption. In spite of expensive operations, a lot of researches still seek to apply AKS in the context of the IoT. The proposed approaches can be classified into two categories: the first category is key transport based on public key encryption, which is quite similar to the traditional key transport mechanism, the category requires from the public key to security transport information. Various key establishment techniques have been proposed for the IoT, ranging from raw public key usage to complex implementations in X.509 standard [Uahhabi et al., 2016]. The second category however, is a Key agreement based on asymmetric techniques, in which a shared secret is derived among two or more parties. In this category, we notice obviously the Diffie-Hellman (DH) key exchange protocol [Qin et al., 2017], and its variants are obviously noticed and will be mentioned later in this chapter.

- Symmetric Key Pre-distribution Schemes: For asymmetric approaches, researchers propose also multiple techniques using symmetric key establishment mechanisms in the IoT security. Symmetric approaches often assume that nodes involved in the key establishment share common credentials. The pre-shared credentials might be a symmetric key or some random bytes flashed into the sensor before its deployment. This category can be divided into two main sub-categories: The first category is probabilistic key distribution. In other words, it is the mechanisms that distribute security credentials (keys or random bytes) chosen randomly from a key pool to constrained nodes [Saikia et al., 2016]. During their initial communication, each two nodes may discover a common key, with certain probability, to establish a secure communication. The second category is the deterministic key distribution. In this sub-category, a deterministic design is applied to create the key pool and to distribute uniformly the keys as such each two nodes share a common key [Sharma et al., 2016].

2.12.1 Asymmetric Key Schemes

The position of asymmetric cryptography or PKC is clear in the conventional Internet. However, it is not the case in the context of the IoT because of its expensive encryption and verification operations [Christianah et al., 2014]. Nevertheless, the development and implementation of PKC in the IoT have never been stopped. In fact, new improvements of several primitives, i.e., Elliptical curve cryptography (ECC) and NTRU continue to reduce the cost of cryptographic operations, so the PKC approach is of a growing interest for constrained environments [Jayapandia et al., 2016]. A brief study in the following sections demonstrates various possible forms of asymmetric key schemes in the IoT.

2.12.1.1 Key Transport Based on Key Encryption

This sub-category looks into the key establishment schemes where the public key is used to transport secret data or to negotiate a session key. Several methods are used to generate the pair of public and private keys [Walsh, 2016]. These mechanisms are classified based on the public/private keys generation methods.

Figure 2.12 gives an example of a communication scenario between two entities A and B. In this scenario, A and B can directly use the public keys to create an encrypted channel. The Certificate Authority (CA) may not be needed to verify the identity of the message transmitter even when the certificates are supported. It should be noted that this method can be expensive for resource-constrained-sensor nodes, in particular when using a traditional algorithm like RSA [Giridhar et al., 2016]. Without a verifiable relationship between the public key and the identity, (i.e., ID-based cryptography, cryptographic-based ID or with CA mediation), this approach becomes vulnerable to the Man in the Middle Attack (MitMA). Indeed, both A and B cannot authenticate each other's identity. An attacker may generate any public/private keys and pretend to be A when communicating with B. Many security protocols have been proposed to secure the IoT networks, [Cao et al. 2015] has proposed an authentication and mutual key establishment scheme for IP based wireless sensor network (6LoWPAN). The authentication has been achieved in 6LoWPAN via Eliptic Cure Cryptosystem (ECC), although, public key cryptography is costly in terms of WSN (6LoWPAN) as shown in [Mstafa et al., 2017], [Baker et al., 2016] and [Haripriya et al., 2016].

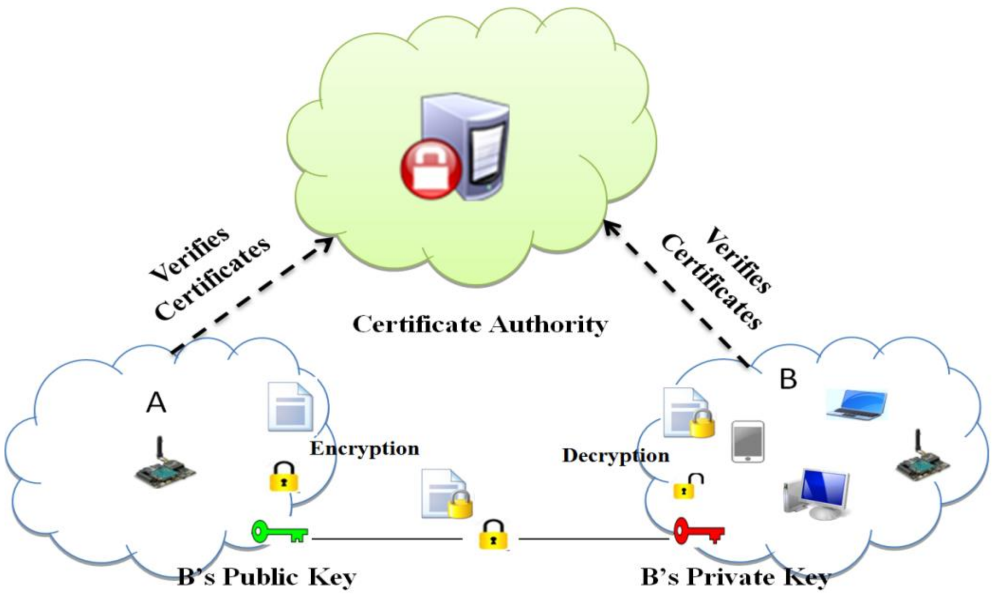


Fig 2.12 Public Key Transport Mechanism [Stallings, 2011]

A- Raw Public key encryption

Some mechanisms assume that the public key has been distributed beforehand or has been used out-of-band communications [Huany et al., 2017]. These mechanisms offer a small number of message exchanges but they are not scalable, because the public keys of all devices should be known by each device. Some ‘‘raw public key encryption’’ mechanisms, i.e., NtruEncrypt [Yin et al., 2014] have been recommended for WSNs. The author in [Nimala et al., 2016] has presented very similar approach to the RSA algorithm (widely used cryptosystem), which is also based upon the hardness of the factorization problem. Furthermore, the scheme requires the same energy consumption for decryption operations as RSA with the same security level [Yajam et al., 2016]. However, it offers much faster mechanism for encryption operations because only one squaring is required to encrypt a message. NtruEncrypt is a cryptosystem which is known to be a lattice-based alternative to RSA and ECC (Elliptic Curve Cryptography) primitives [Mstafa et al., 2017]. The mechanism is highly efficient and suitable for the most limited-resource devices such as smartcards and RFID (Radio-Frequency Identification) tags. In [Bafandehkar et al., 2013], the author gives a comparison of the three PKC mechanisms proposed for constrained devices: The author NtruEncrypt and ECC. The results show that NtruEncrypt leads to the smallest average power consumption. However, this cryptosystem often requires large-size messages, and might result in packet fragmentation at lower layers and many re-transmissions in the presence of communication errors [Liu et al., 2016].

B- Certificate-based Encryption

Certificate-based protocols are a popular choice to establish a secure communication between two entities over Internet. The trust relationship between the two entities is guaranteed by a well-known third party (CA) using the standard X.509 certificate that validates the identity of the entity as Figure 2.12 shows. Indeed, each sensor node possesses a certificate signed by the trusted CA. The latter can be loaded into the node before the deployed node directly requested from a trusted party. Transport Layer Security (TLS) [Yu et al., 2015] has been recommended by many standards specified by IETF for security services. However, it is mentioned in [Bafandehkar et al., 2013], that TLS is not a wise choice with respect to the security best practices in the IoT. In fact, TLS runs normally in a reliable transport protocol like TCP which is unsuitable for constrained resource devices, due to its congestion control algorithm. As a replacement for TLS in the tightly constrained environments, the Datagram Transport Layer Security (DTLS) protocol has been proposed recently [Kothmay et al., 2013]. It operates over the unreliable transport protocol, i.e., UDP and provides the same high security levels as TLS. The utilization of a certificate is basically expensive.

C- ID-based Cryptography (IBC)

The IBC is cryptographic scheme that was first proposed by [Tan et al., 2016]. The scheme enables users to communicate securely and verify each other's signature without exchanging public or private keys as shows in figure 2.13. However, the scheme requires the presence of Trusted Key Generation (TKG) centres demonstrated as the following:

The IBC's Operation is unlike the normal Public Key Infrastructure (PKI) where a TKG randomly generates pairs of public/private keys, each node in IBC chooses its identifier (address or name) as a public key. Practically, any public known information that uniquely identifies the node could be used as a public key. The TKG generates the corresponding private key and security distributes it to the node.
As figure 2.13 shows, when node (A) wants to communicate with another node (B), node A will sign the message using its private key and encrypt the result with the node B's public key. Upon receiving the message, node B will decrypt the message using its private key and verify the signature using node A's public key.

The IBC represents an efficient and an easy system to be implemented which removes some of the overheads encountered in PKI for key management and digital certificate issuance/ revocation. However, the security of the IBC is based on the security of the private key [Wang et al., 2016]. To deal with this issue, the node needs to combine additional information such as timestamps to their identifiers when generating the public key. This procedure will definitely guarantee a periodic update of the public key. However, it introduces a key-management problem where all users must have the most recent public key for the node.

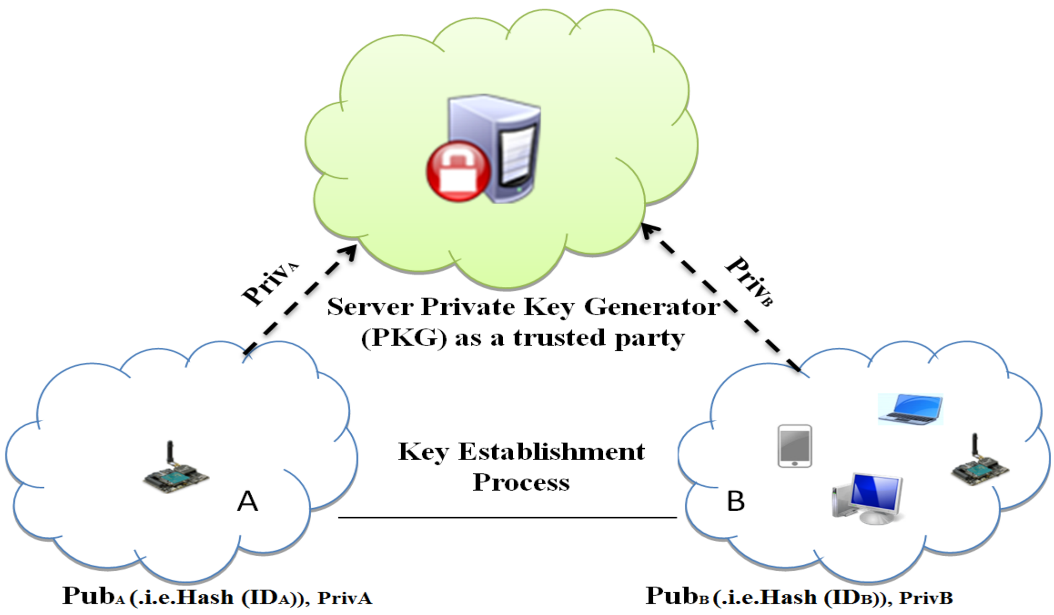


Fig 2.13 Identity-based cryptography infrastructure [Stallings, 2011]

In a constrained environment, ID-Based Encryption (IBE) model is mostly implemented using the ECC primitive [Fanian et al., 2010]. Implementations on other primitive exist, for example, RSA or El-Gamal-type IBE [Mikhail et al 2014]. [Yang et al., 2013] has proposed Identity-based-Authentication and Key Agreement (IBAKA) – an IBE scheme inspired by [Tan et al.2016]. However, they combined the IBE method with the Elliptic curve Diffie-Hellman (ECDH) [Qin et al., 2017] key exchange in order to establish a session key.

2.12.1.2 Key Agreement based on Asymmetric Techniques

This sub-category is about key agreement protocols based on asymmetric primitives in the IoT. As mentioned in various research works, a key agreement protocol is the mechanism where two (or more) parties derive a shared secret and no other party can predetermine the secret value. Figure 2.14 illustrates the process of a typical asymmetric key agreement. Km is the secret generated after the agreement procedure. Consequently, this symmetric key is then used to secure the communication.

The Diffie–Hellman (DH) protocol and its variants are classical examples for symmetric key agreement [Mortazavi et al., 2011]. However, the DH protocol is vulnerable to different attacks such as DoS and MitMA; thus, using this protocol can affect the security performance of the IoT devices. On the other hand, some variants of the DH protocol are considered in constrained environments using ECC, i.e., ECDH. The ECDH cryptographic primitive offers smaller key size than RSA. Indeed, the US National Institute for Standard and Technology (NIST) in [Soliman et al., 2016] has showed that to achieve the security level of 128-bit AES key size, one can prefer 256 bit key size using elliptic curve instead of 3072 bit parameters in RSA and DH protocol. The scheme relies on the ECDH protocol, and additionally provides the privacy of message exchanges using identity-based scheme [Yao et al., 2015].

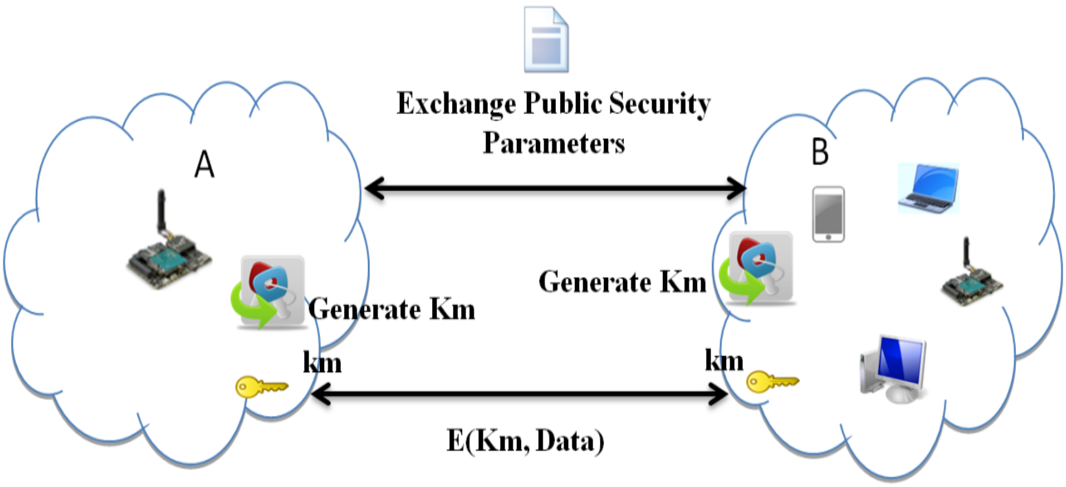


Fig 2.14 Key Agreement on Asymmetric Mechanisms [Stallings, 2011]

Host Identity Protocol Diet Exchange (HIP-DEX) [Sahraui et al., 2014] applies also the DH protocol to generate a session key between two entities after only a 4 message exchange. This protocol is a variant of HIP Base Exchange [Takahashi et al., 2012] designed, specially, to reduce the complexity of cryptographic computations. It uses the smallest possible set of cryptographic primitives (e.g. AES-CBC instead of cryptographic hash functions), removes digital signatures and implements static ECDH to encrypt the session key, etc. This protocol has been largely taken into consideration in the context of the IoT by many recent works [Yang et al., 2013]. For example, [Meca et al., 2013] proposed an efficient network access mechanism based on HIP-DEX for mobile nodes joining the local sensor network. Besides, [Hummen et al., 2013] tailored HIP-DEX to the IoT, in particular, by adapting the session resumption mechanism as in TLS [Paris et al., 2017]. As such, the constrained node performs expensive operations once and maintains session-state for re-authentication and re-establishment of a secure channel. The key agreement protocols based on DH require fewer messages to establish a session key but the computational tasks on sensor nodes are usually complex.

2.12.2 Symmetric Key Pre-Distribution

In this sub-category, the key pre-distribution mechanism is divided into two main sections as the following:

2.12.2.1 Probabilistic Key Distribution

The mechanism of Random Key Pre-distribution (RKP) was first proposed by [Miyaji et al., 2013]. A typical RKP consists of three phases: key pre-distribution, shared-key discovery and path-key establishment. In the scheme, a large key pool is generated. Keys are then randomly selected from the key pool and distributed to sensor nodes. Any two nodes may share a common key with a certain probability. The third phase is triggered when two nodes do not share any common key. In this process, one node first generates a random key K. It then sends the key to its neighbours using the pre-established secure channel. The process continues until the key K arrives at the other node. K is considered afterward as the pairwise key between both nodes. Several solutions are inspired by this scheme [Papadimitratos et al., 2012].

These proposals improve specially the pre-distribution phase to enhance the key connectivity between nodes and reduce the memory space needed for key storage. In fact, [Liu et al., 2006] proposed a key pre-distribution scheme that relies on the deployment knowledge and avoids unnecessary key assignments. On the other hand, [Mehmood et al., 2017] developed a scheme based on [Reegan et al., 2016] works but the keys are mapped on two-dimensional positions. Both of them proposed a probability density function which provides a better key connectivity. [Levi et al., 2017] developed also a mechanism to reinforce the path-key establishment phase. The basic idea is that node A finds all possible links to node B. It generates for each link a random value and routes these values to B. The common keys between A and B are protected by these random values. However, the generated key will be shared by both nodes, unless the adversary manages to eavesdrop on all paths between them. The probabilistic key distribution generally does not guarantee session key establishment between all nodes even with the path-key establishment phase. Two nodes may not share any common keys with a certain probability.

2.12.2.2 Deterministic Key distribution

In this sub-category, the described key schemes rely on a deterministic process to generate the key pool and to distribute keys to nodes in order to guarantee secure full connectivity in the network. In deterministic solutions, the key schemes are distinguished by either the presence of a trusted third party in the IoT security or not.

A- Offline key Distribution

The offline key distribution method is used widely in WSNs because of its simplicity. Depending on the used protocol, every node in the same network may share a network key or each two nodes may have a common pairwise key [Tong et al., 2013]. The session key is then generated after very few data exchanges without the presence of any third party. The offline key distribution provides efficiency in terms of energy consumption because it does not require expensive cryptographic computations like asymmetric approaches. However, when a sensor node is physically attacked, the secret data stored inside the node can be exposed. Consequently, the attacker can gain access to several nodes which share the secret key with the attacked node, or in the worst case, it may access the whole network [Alejandro et al., 2016].

In several existing works, mathematical properties have been applied to create the model for securing key exchanges between sensor nodes. These mechanisms are still applicable in the context of the IoT. The most well-known schemes are based on bivariate polynomials [Klodowski et al., 2016]. In these schemes, node A shares with other nodes a bivariate n-degree polynomial f(x,y). Node A can obtain the pairwise key with another node B by calculating the value of f(IdA, IdB), where IdA and IdB are the respective identities of A and B. In the same way, B can obtain the same pairwise key, since f(IdA, IdB) is equal to f(IdB, IdA). In another scheme, called the Bloom's scheme [Rescorla, 1999], a secret symmetric matrix D is generated from the shared secret key between two nodes A and B. Each of them generates a public matrix IA and IB respectively for A and B. The private keys are respectively privA = D x IA and privB = D x IB for A and B. Finally, the pairwise key is calculated by solving (privA x IB) or (privB x IA). However, the problem with these latter two schemes is that the session key will remain unchanged for every two nodes [Klodowski et al., 2016].

SNAKE and BROSK [Yeh et al., 2016] are two key establishment schemes where the session key is generated without the need for a key server to perform key management. These two protocols assume that all nodes in the same network share a master secret key. In SNAKE, the session key is obtained by hashing two random nonces generated from each communicating party using the pre-shared key. BROSK broadcasts the key negotiation message containing a nonce. Once a node receives the message from its neighbours, it can construct the session key by computing the message authentication code (MAC) of two nonces. [Driessen et al., 2012] has investigated the ability to secure the communication for smart IoT objects. The objective of this work is to design a lightweight protocol procedure to set up secure end to end channels between unconstrained and remote peers and IoT devices. The author addressed security in terms of resilience against node capture via using lightweight IPsec security association [Driessen et al., 2012]. Furthermore, AH and ESP mechanisms provide origin authenticity, message integrity and confidentiality protection of IP packets but they do not handle the key exchange. The security associations are established manually using pre-shared key and, the offline key distribution does not provide rekeying operations. When the system changes to other secret keys; all the entities in the network need to be updated to establish secure communications using the new keys.

B- Server Assisted key Distribution

Due to the resource limitation of constrained devices, the cryptographic computation and other expensive tasks e.g. identity management and key generation, can be handled at rich–resource servers. Server-assisted approaches for key establishment protocols have been proposed in this respect in IoT. In such protocols, message exchanges engage two entities and one (or more) trusted servers. The server shares long-term key a priori with each communicating entity. It often plays the role of a Key Distribution Center (KDC) and then supplies the session key to each party by re-encrypting it using the shared keys as shown in Figure 2. 15.

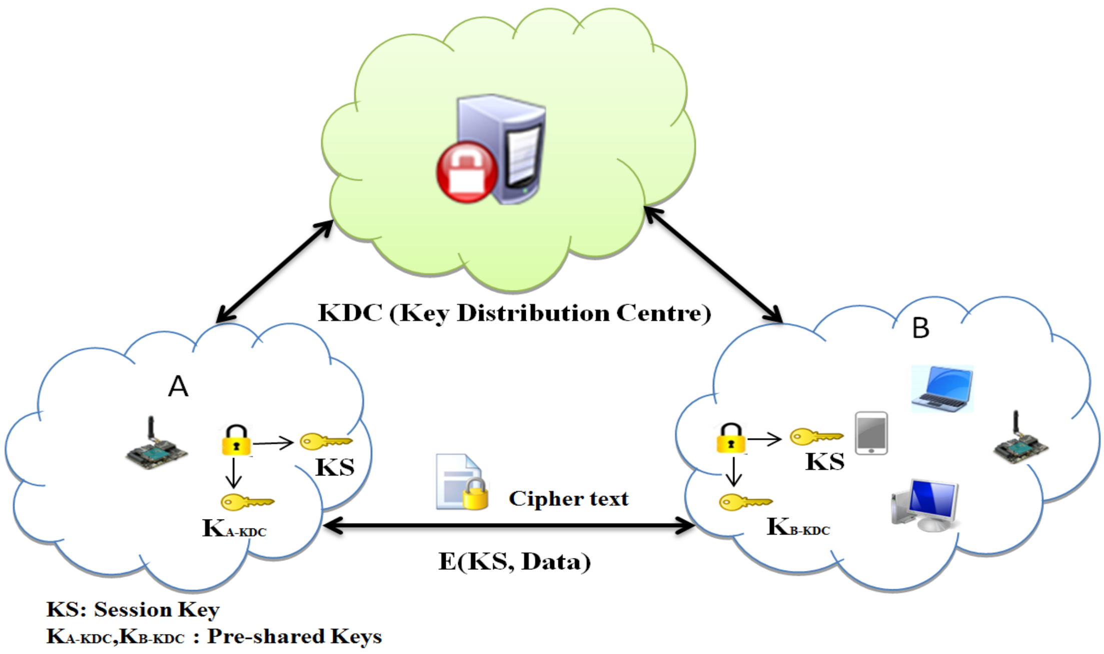


Fig 2.15 Server-Assisted Mechanism [Stallings, 2011]

MIKEY [Boudguiga et al., 2013] is a protocol, in which a KDC is involved in the process of establishing a security association between the two parties. MIKEY-Ticket originated from the ticket concept of Kerberos [Zhang et al., 2016], [Prakasha et al., 2016], [Tbatou et al., 2015]. The KDC securely communicates with the node initiating the protocol (Initiator) and the responding node (Responder) by encrypting important data using the pre-shared master key shared with each node. Nevertheless, the protocol is vulnerable to DoS attacks, particularly replaying messages to the Responder. To prevent these attacks, [Bouduguia et al., 2013] has proposed a new key establishment, called Sever Assisted Key Establishment (SAKE) based on the MIKEY-Ticket mode but removing the threat of DoS attacks. SAKE [Hussen et al., 2013] allows establishing security associations between the two parties after only five exchanged messages, compared to six messages in the original MIKEY-Ticket. Indeed, upon reception of the first message from the Initiator, the KDC generates the session key and contacts directly the Responder. This change reduces one message exchange comparing to MIKEY-ticket.

Protocol for Carrying Authentication for Network Access (PANA) runs over UDP and uses Extensible Authentication Protocol (EAP) [Pawlowski et al., 2015] for authentication that supports multiple authentication methods including pre-shared key distribution. [Hernndez-Ramos et al. 2015], [Bernal-Hidalgo et al., 2014] propose an improvement of PANA to adapt the resource-constraints. The main modifications consist of reducing the number of message exchanges (e.g. choosing EAP-PSK as the only authentication method), removing unused PANA header fields, minimizing the collection of cryptographic primitives at the constrained device [Forsberg et al 2015]. These proposals may reduce effectively the PANA implementation code size at the device; however, the authors do not show that reduces the number of messages could affect the strength of security protocol which can make it easy to capture the IoT device from the adversary.

2.13 Verifying Security Protocols

Protocols verification and validation can be achieved using different approaches. Discrete event simulators such as NS2 [Hossain, 2009] and OPNET [Aboelela, 2007] provide good capabilities to analyse the protocol performance. Other approaches based on the Unified Modelling Language (UML) or the Specification and Description Language (SDL) provide validation methods to check the protocol against its specification in order to prevent undesired states and behaviour. This includes preventing deadlocks and live locks. However, the verification of security protocols against their claimed properties requires special toolsets as mathematical logic or model checks. In general, verifying security protocols is based on theorem proofs and verification logic such as the BAN logic [Burrows et al., 1990] which determines the trust relationship among the protocols' parties. However, the BAN logic considered the authentication properties only, therefore, it could not be used in confidentiality analysis. Also, the BAN logic assumed all parties to be honest and trustworthy, thus this assumption has to be considered when reading the BAN results. In this research, AVISPA tool is used to verify security protocols implementation. AVISPA is a push tool for the automated validation of security protocol [AVISPA, 2013]. In other words, significantly, a modular and expressive formal language called High level protocols specification language (HLPSL) is used by AVISPA to specify the security protocol and their properties [AVISPA, 2013]. HLPSL is a role-based language, meaning that the sequence of actions of each kind of protocol participant in a module should be specified first; this is called a basic role. This specification can be later presented by one or more agents that play the given role. Later on, this document will specify how the resulting participants interact with one another by combining multiple basic roles together into a composed role.

HLPSL specification is translated into the Intermediate Format (IF), using hlpsl2if. The IF specification is then processed by the model-checkers to analyse if the security goals are violated or not. As figure 2.16 shows, there are four different verification back end tools that are used to analyse the IF specification namely: On-the-Fly Model-Checker (OFMC), Constraint-Logic-based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC) and Tree Automata-based Protocol Analyser (TA4SP). Possible flaws in a protocol can be identified using these back end tools. As exponential and XOR operations are supported by CL-AtSe and OFMC back ends, OFMC back end tool will be used with AVISPA and SPAN (Animation tool for AVISPA) to analyse the proposed protocols [AVISPA, 2013].

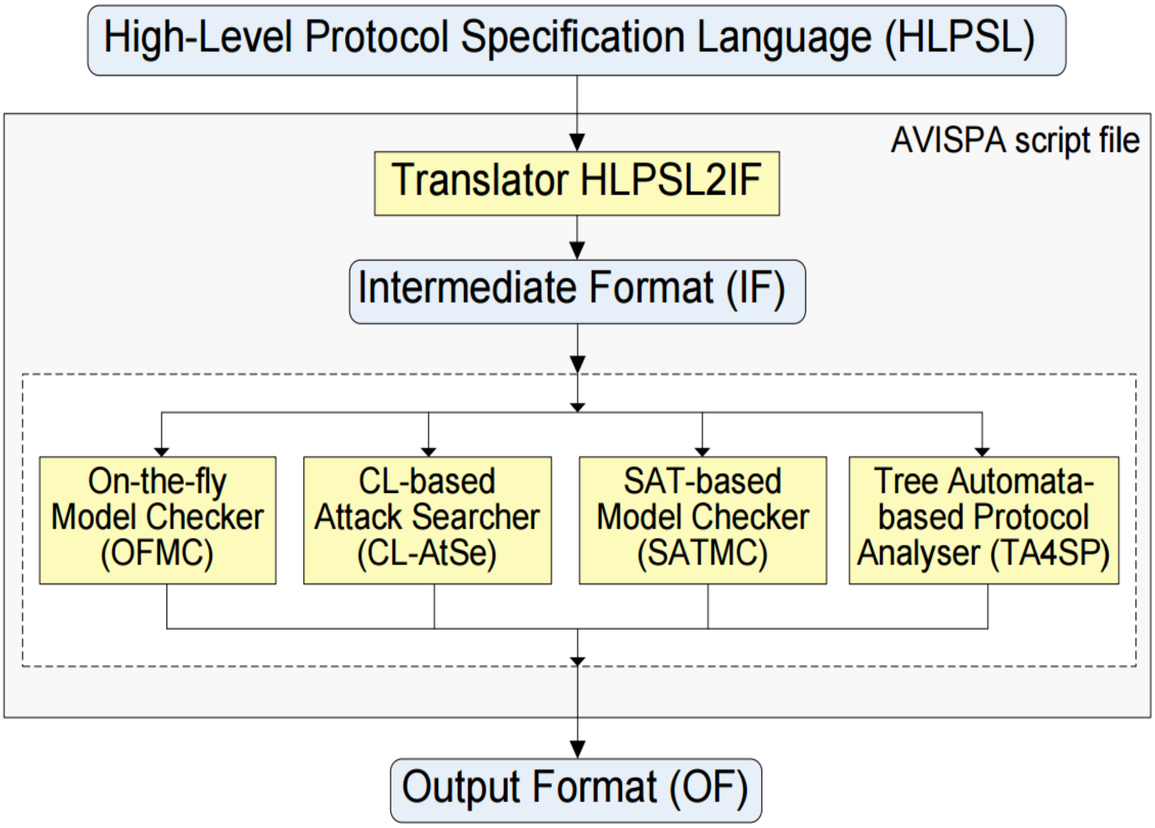


Fig 2.16 Architecture of the AVISPA tool [AVISPA, 2013].

In order to test and evaluate the security performance protocol, a Contiki and Cooja simulation tool is used [Contiki, 2014]. It is to be mentioned here that the Contiki and Cooja is an open source operating system for the Internet of Things. It connects tiny low-cost and low-power microcontrollers to the Internet. Indeed, it is a powerful toolbox used for building complex wireless systems [Contiki, 2014].

2.14 Summary

In this chapter, different types of IoTs/6LoWPAN Communication and mobility protocols are discussed. As shown below, the existing protocols in the IoT such as MIPv6, PMIPv6, and NEMO are reviewed. With the knowledge of the 6LoWPAN network, it has been demonstrated that the Edge Router uses the NEMO extensions of Mobile IP. Some comparisons of the mobility protocols have been done in terms of different mobility and communication scenarios. The exchange messages used for neighbour discovery in IoTs/6LoWPAN networks have also been investigated. As shown in table 2.1; Networks architecture comparison, each mobility protocol has its own advantages depending on the scenarios it involves.

Table 2.1 Network Architecture Comparison

Network Architecture	Router Size	Concretively	Security protocol
MIPv6 [Imran et al 2016]	Small	Limited	Yes
PMIPv6 [Huang et al., 2017].	Small	Medium	Yes
NEMO [Ye et al., 2017]	Small	Medium	Yes
LISP [Cisco-A, 2014]	Big	Good	No

However, most of these mobility protocols, such as MIPv6, PMIPv6 and NEMO, might be too much for low-bandwidth wireless links in domains with large LoWPAN. Adding to this, as the number of connected devices increases, i.e., IoT devices, the burden on the network infrastructure increases as well. One of the key challenges will be the size of the routing tables and efficiency of the current routing protocols in the Internet backbone. For that reason, LISP supports different types of networks, i.e., IoT, which defines compressed and size optimised mobility/communication signalling. Though, as mentioned earlier the LISP protocol and the security protocol, in particular, are still at an early stage of implementation.

As shown above this chapter has provided therefore a comparative classification of existing protocols for IoTs. These protocols and techniques are analysed according to different criteria in order to identify the advantages and drawbacks of each protocol as table 2.2 shows.

Using this methodology, it was noted that symmetric approaches are not anymore the default choice for IoT of robust communications security between the entities. Public key cryptography is increasingly recommended in the IoT context, provided that the associated asymmetric techniques are properly optimized. A trusted third party will certainly take a more active role to secure the IoT and adapt to its heterogeneous nature.

Table 2.2 Security Protocols Comparison

<table>
<tr><th colspan="7">Security Protocol in Internet of Things</th></tr>
<tr><th colspan="4">Asymmetric key</th><th colspan="3">Symmetric Key</th></tr>
<tr><td colspan="3">Key Transport based on Public Key Encryption</td><td>Key Agreement on Asymmetric Techniques</td><td>Probabilistic Key Distribution</td><td colspan="2">Deterministic Key Distribution</td></tr>
<tr><td>Raw Public Key Encryption</td><td>Certificate based Encryption</td><td>Identity-based Schema</td><td rowspan="2">Diffie-Hellman IBAKA HIP-DEX

[Qin et al.,2017]</td><td rowspan="2">Random key Schemes [Rahman et al., 2017]</td><td>Offline key</td><td>Server-Assisted Key Distribution</td></tr>
<tr><td>Protocols based NTRU Encrypt

[Yin et al., 2014]</td><td>DTLS

[Kothmay et al., 2013</td><td>Tiny IBE IBAKA [Masdri et al., 2017]</td><td>Bivariate Ploynomail Blom’s scheme Based SNAKE, BRoSK [Yeh et al., 2016]</td><td>Kerberos Mikey-ticket

[Boudguiga et al., 2013]</td></tr>
<tr><th colspan="7">Security Benefits</th></tr>
<tr><td colspan="4">-In asymmetric or public key, cryptography doest not need exchanging keys, thus eliminating the key distribution problem.</td><td colspan="3">-A symmetric cryptosystem is faster</td></tr>
<tr><td colspan="4">-The primary advantage of public-key cryptography is increased security: the private keys do not ever need to be transmitted or revealed to anyone.</td><td colspan="3">-In Symmetric Cryptosystems, encrypted data can be transferred on the link even if there is a possibility that the data will be intercepted. Since there is no key transmitted with the data, the chances of data being decrypted are null.</td></tr>
<tr><td colspan="4" rowspan="2">-Can provide digital signatures that can be repudiated</td><td colspan="3">-symmetric cryptosystem uses password authentication to prove the receiver’s identity</td></tr>
<tr><td colspan="3">- Only symmetric cryptosystem possesses the secret key and it can decrypt a message.</td></tr>
<tr><th colspan="7">Security limitations</th></tr>
<tr><td colspan="4" rowspan="2">-Security limitation of using public-key cryptography for encryption is speed: there are popular secret-key encryption methods which are significantly faster than any currently available public-key encryption method.</td><td colspan="3">-Symmetric cryptosystems have a problem of key transportation. The secret key is to be transmitted to the receiving system before the actual message is to be transmitted. Every means of electronic communication is insecure as it is impossible to guarantee that no one will be able to tap communication channels. So the only secure way of exchanging keys would be exchanging them personally.</td></tr>
<tr><td colspan="3">- Symmetric cannot provide digital signatures that cannot be repudiated</td></tr>
</table>

Added to this, security protocols should take into account the resource-constrained feature of things. Heavyweight cryptographic operations based on RSA and public key cryptography should be replaced by lightweight operations, i.e., using symmetric cryptography or applying more lightweight asymmetric primitives [Lara-Nino et al., 2017]. Besides, lightweight security protocols are also needed to reduce the communication complexity. Aside from performance concerns, the future proposed security solutions will offer perspectives on new applications that increasingly expand the coverage of capabilities and features offered by the IoT devices making them more and more intelligent. However, reducing the exchange messages in security protocols could break the security and capture the IoT devices by adversary. As the number of communication used in IoT device is increasing, designing a security protocol to be implemented will be the biggest challenge, especially with the deployment of a huge number of devices on the network. Therefore, a new security protocol is needed to secure the IoT devices, which can provide End-to-End secure communication process between IoT nodes. This security protocol should be adopted in accordance with the nature of IoT devices and infrastructure of the networks. Therefore, the next chapter will provide a comprehensive security threats/attacks analysis using X.805 framework for IoT based on LISP network architecture.

Chapter 3:

Security Issues and Analysis in Internet of Things based on LISP Network Architecture

3.1 Introduction

To begin with, the importance of thethe Interenet of Things (IoT) les in the fact that integrates various sensors devices that enable communication with each other without human interference. In this regard, the IoT concept trefers to the usage of standard Internet protocols that allows communication among those devices. In other words, a Locator ID Separation Protocol (LISP) is a routing architecture that provides new IP addressing in order to simplify routing operations and improve scalability in the Future of the Internet such as the IoT. Because, LISP and the security protocol are still under development, this chapter investigates the security issues that could occur from deploying the Locator ID Separation Protocol in the IoT. The investigation discovers a number of vulnerabilities that should be considered before moving to the implementation stage. Accordingly, the structure of this chapter is divided as the following: In section 3.2, the current security threats and attacks in the IoT are tackled. In section 3.3, a brief overview of X.805 framework is given. In section 3.4, a comparative security analysis via using X.805 security framework is provided. The aim of the whole chapter, however, is to analyse the security performance of the IoT based on LISP network architecture. The results of the analysis are summarized in section 3.5.

3.2 Security Threats in Internet of Things

Most of the attacks and threats against devices and data security in IoT have a destructible effect, because of their wireless radio access and connectivity to the Internet. The security analysis of IoT starts with the appreciation of various threats posed at respective Open System Interconnection (OSI) layers. In this section, the threats in IoT network are classified and discussed [Husamuddin et al., 2017].

The IoT is highly vulnerable to physical attacks, i.e., threats due to physical node destruction, and relocation. For example, an intruder or attacker can redirect all packets between two nodes to himself. By sending, on the one hand, spoofed location update to the gateway from Sensor-A to the new location of Sensor-B indicating its own locator (intruder) for the new location and them, sending other spoofed location update to gateway from the Sensor-B about the new location of Sensor-A indicating also the new location to its own locator. Thus, the intruder or adversary is able to capture the data between Sensor-A and Sensor-B, establish new spoof connections as Sensor-A and/or Sensor-B, and finally insert itself in the middle of all connections between them (man in the middle attack). Therefore, the intruder receivers and can modify all the packets exchanged between Sensor-A and Sensor-B [Khan et al., 2016]. Moreover, and many of these attacks allow the malicious node to take control over them. These compromises can result into code modification inside the node and change in the role of networks and sensors. In addition, several types of DoS attacks can be triggered in different IoT and 6LoWPAN environment and in different layers. At the physical layer, the DoS attacks can be launched by tampering and jamming electromagnetic (EM) signals and by swarming the limited resources of 6LoWPAN devices with the high resource devices quite easily.

An attack on MAC layer involves collision, exhaustion and unfairness. Being always power hungry, 6LoWPAN devices try to sleep as often as possible in order to keep it. Such constraints allow the attacker to let the device execute a large number of tasks in order to deplete its battery. This is called sleep deprivation torture [Nawir et al., 2016]. To achieve such a goal, an attacker can, for example, target different destination devices with unnecessary packets, possibly in other 6LoWPAN, regardless of whether the destination 6LoWPAN and/or device actually exists or not. Such attack can also lead to deplete the 6LoWPAN coordinator battery power. In other words, the downlink packets have to be clearly requested from the LoWPAN coordinator; this will in turn keep it busy.

An attack against network availability can consist of flooding the network by simply transmitting a large number of large packets. In such a case, the attacker may degrade the network performance and reduce the throughput in general. In WPAN specification, the replayed message is prevented by the replay protection mechanism, i.e., sequential freshness. In a replay-protection attack, the malicious node sends many frames containing large counters to a particular receiver, which in turn raises the replay counter up [Aris et al., 2015]. Then, when a normal device sends a frame with a lower frame counter, it will be rejected by the receiver and thus, leads to DoS attack.

As the ACK frame integrity is not protected, it can possibly open the door for a malicious node to prevent a legitimate device from receiving a particular frame. This is possible by forging an ACK using the unencrypted sequence number from the data frame and sending it to the source while creating enough interference in order to prevent the legitimate receiver from receiving the frame [Nurse et al., 2015]. In such scenario, the source device leads the belief that the frame has been received. Moreover, a corrupted device can also attack the key distribution process since the WSN coordinator announces the IDs of devices that can change the link key in plain-text in the beacon frame. Therefore, the attacker can send request packet with the ID of the legitimate node. Obviously, the goal from such request is to push the coordinator to trigger a key exchange process while the legitimate recipient may not be ready [Yang et al., 2016]

3.2.1 Attacks Against Network Layer

As stated earlier, this thesis focuses on designing a security protocol to the IoT at network layer; it is therefore very important to disclose attacks on this layer. The following types explain the attacks of IoT in a network layer [Dragomir et al., 2016]:

- Spoofing: in this attack, the malicious node uses spoofing to target routing information exchanged between nodes in an attempt to create routing loops attack or repel network traffic extending/shortening source routes, generating false error messages, etc.
- Selective forwarding: in this attack, the malicious device may refuse to forward certain messages by dropping them, for instance. In this case, neighboring devices may conclude that the malicious device has failed and thus tries to seek another router. A more subtle form of this attack is when the malicious device selectively forwards packets. However, neighboring nodes here will not be able to reach the conclusion that another route is needed. This would in turn encourage them to resend the data packets.
- Sinkhole attack: in a sinkhole attack, the malicious device tries to get all traffic from one particular area which can potentially result in DoS attack. In order to launch a sinkhole attack (black hole attack), the attacker can listen to requests for routes then replies to the requesting nodes that contain high quality or the shortest path to the base station. Once the malicious device is able to insert itself between the communicating nodes, he/she is able to do anything with the packets passing through it. In fact, this

attack can affect even the nodes that are located farther from the malicious node [Dragomir et al., 2016].

- Sybil attack: in a Sybil attack, a single node presents multiple identities to other nodes in the IoT/ or WPAN. Sybil attacks pose a significant threat to geographic routing protocols and may be performed against the distributed storage, routing mechanism, data aggregation, and voting, fair resource-allocation and misbehaviour detection.

- Wormhole attack: in a wormhole attack, the attacker records packets at one location in the network and tunnels them to another one. Such attacks can damage the working of the 6LoWPAN since it does not require compromising a node in the WPAN. Instead, it could be performed at the initial phase when 6LoWPAN nodes start to discover the neighbouring information. Wormhole attacks can target, for example, routing function or application.

3.2.2 Analysis

Essentially, IPsec works well on non-Low-power devices which are not subject to severe constraints on host software size, processing and transmission capacities. Furthermore, IPsec supports AH for authenticating the IP header and ESP for authenticating and encrypting the payload. The main issues of IPsec are twofold: firstly, processing power and, secondly, key management. Since these tiny IoT devices do not process huge number of data or communicate with many different nodes, it is not well understood if complete implementation of Security Association Database (SADB), policy-debase and dynamic key-management protocol are suitable for these small battery powered devices. Besides, given existing constraints in IoT environments, IPsec may not be suitable to use in such environments, especially that IoT node devices may be able to operate all IPsec algorithms on its own capability either Full-Function Device (FFD) or Reduced-Function Devices (RFD). Bandwidth is very rare resource in IoT environments. The fact that IPsec requires another header (AH or ESP) in every packet makes its use problematic in IoT environments. Moreover, IPsec requires two communicating peers to share secret key that is established dynamically with the Internet key Exchange (IKEv2) protocol. Thus, it has an additional packet overhead incurred by IKEv2 packets exchange. Therefore, a new security protocol is needed to address this issue.

3.3 Overview of X.805 Security Framework

The X.805 standard proposes three security layers: The first security layer is the applications security layer; which are in fact network-based applications accessed by end-users e.g. web browsing, directory assistance, email, and E-commerce. The second security layer is services security layer; which is services provided to end-users e.g. Frame Relay, IP, cellular, Wi-Fi, and Voice over Internet Protocol (VoIP). The third security layer is Infrastructure Security layer; which is Fundamental building blocks of networks services and applications e.g. Individual routers, switches, servers, Point-to-point WAN links and Ethernet links. Furthermore, it has three security planes: End user Plane which is Access and use of the network by the customers for various purposes, e.g. Basic connectivity/transport and value-added services Virtual Private Network (VPN) and VoIP. Control/Signaling Security Plane; which refers to any Activities that enable efficient functioning of the network, e.g. Machine-to-machine communications. Management Security plane; which is the management and provisioning of network elements, services and applications and also Support of the FCAPS functions: acronym for fault, configuration, accounting, performance and security. Finally these security layers and planes are based on the performed activities over the network and eight security dimensions to address general system vulnerabilities (access control, authentication, non-reputation, data confidentiality communication security, data integrity, availability and finally privacy) [Raheem et al., 2013]. Figure 3.1 illustrates the complete architecture of the X.805 standard with its security layers

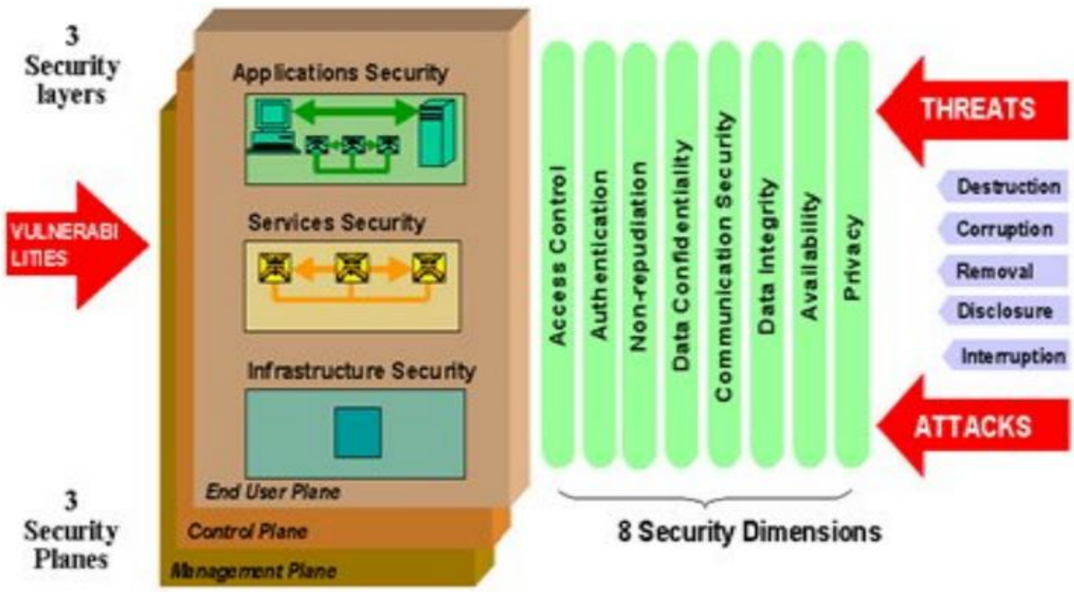


Fig 3.1 The X.805 Standard Architecture [Raheem et al., 2013]

In this section, the X.805 security framework standard has been applied to analyse the security performance of IoT based on the LISP architecture. In addition, the functionality of these devices (IoT) is only related to the Infrastructure Layer and services Layer of the X.805 standard. In LISP architecture, there are two planes only; Control Plane and User Plane [Raheem et al., 2013]. These planes are concerned with the security of the network links and elements as shown in figure 3.2.

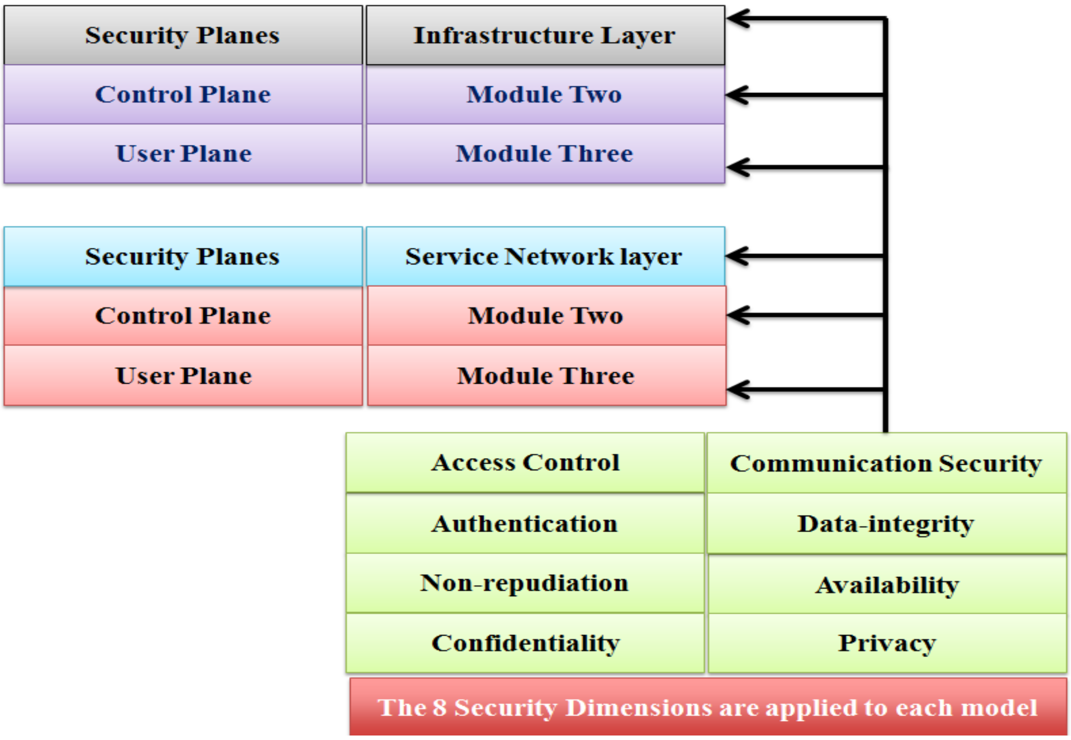


Fig 3.2 X.805 Standard for Internet of Things using LISP Network Architecture

3.4 Security Analysis to Internet of Things using LISP Network Architecture

In this section, a security analysis to the IoT based on LISP network architecture using X.805 security framework is provided.

3.4.1 Access Control

- **Infrastructure Layer:** Modules Control Plane & User Plane: ACLs (Access Control) can be applied in LISP between the Ingress direction and the Egress direction on the LISP site-facing interface. It plays as a packet filtering between two networks.

Therefore, ACLs are needed when one specific network might choose not to receive packets from the other networks or make connections to other networks.

- **Service Layer:** Modules Control Plane & User Plane, ACLs can be applied between an XTR router and the map services, which should be reflected in the registration procedure stage. Therefore, unregistered routers cannot send updates with the mapping system.
- **Threats and Attacks:** Figure 3.3 shows that unauthorized devices/or illegitimate devices can access the network. Added to this, Dos attacks can intercept packets between the (ITR & ETR) and between the XTR router and mapping system. Therefore, in this attack the intruder is able to redirect all the traffic sent by two nodes to a random or non-existent Locator, in order to stop or disrupt communication between the nodes.

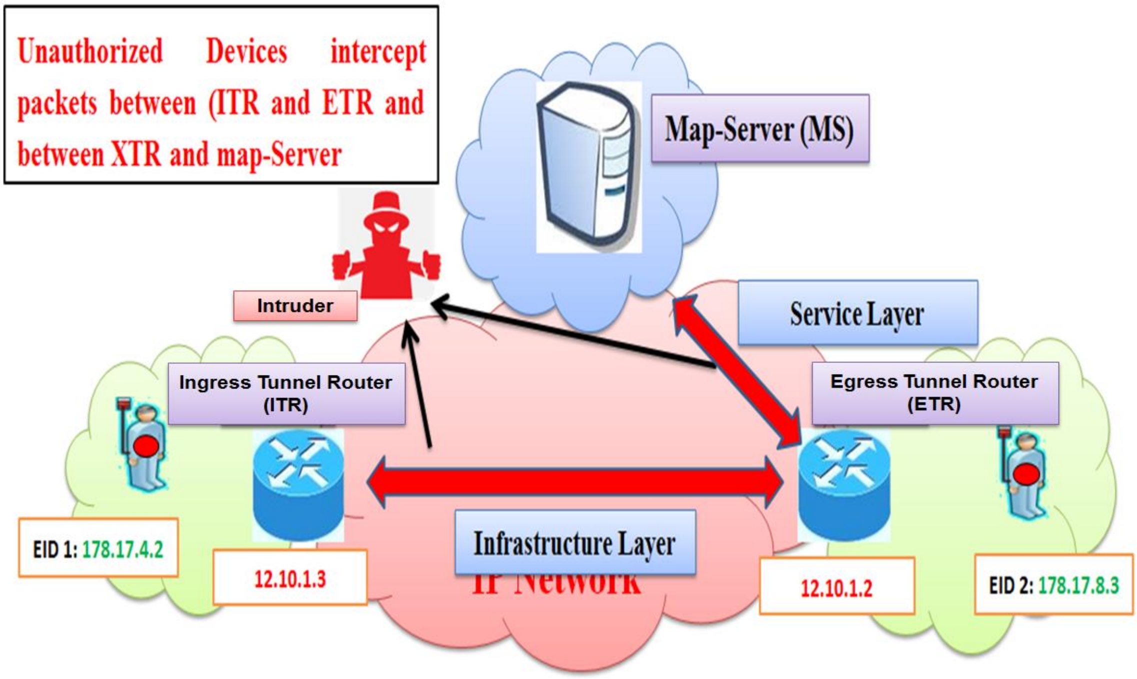


Fig 3.3 Threats in Access Control

3.4.2 Authentication

- **Infrastructure Layer:** Modules Control Plane & User Plane: Users/or machines that register themselves with the XTR router require authentication to ensure that whoever the user or the machine claims to be, it should be the correct one. For example, Authentication needs to be applied when machine (A) wants to communicate with machine (B).

- **Service Layer:** Modules Control Plane & User Plane: When the Router XTR is registered with the mapping system, it needs to be authenticated with another router
- **Threats and Attacks:** As Figure 3.4 shows, the attackers (unauthenticated devices) can access the network by claiming fake identities to the router or the mapping system. Furthermore, the presented attack can be also on ETR Router as figure 3.4 illustrations, where the attacker is able to create a spoofed binding update, this attack appears because ETR router is not authenticated. Besides, the non-authentication of the ETR router presents other kind of attack which can steal the device ID based on spoofed host name registered update message which are originally sent between ETR router and EID 2 device, likewise for ITR router and EID 1 device.

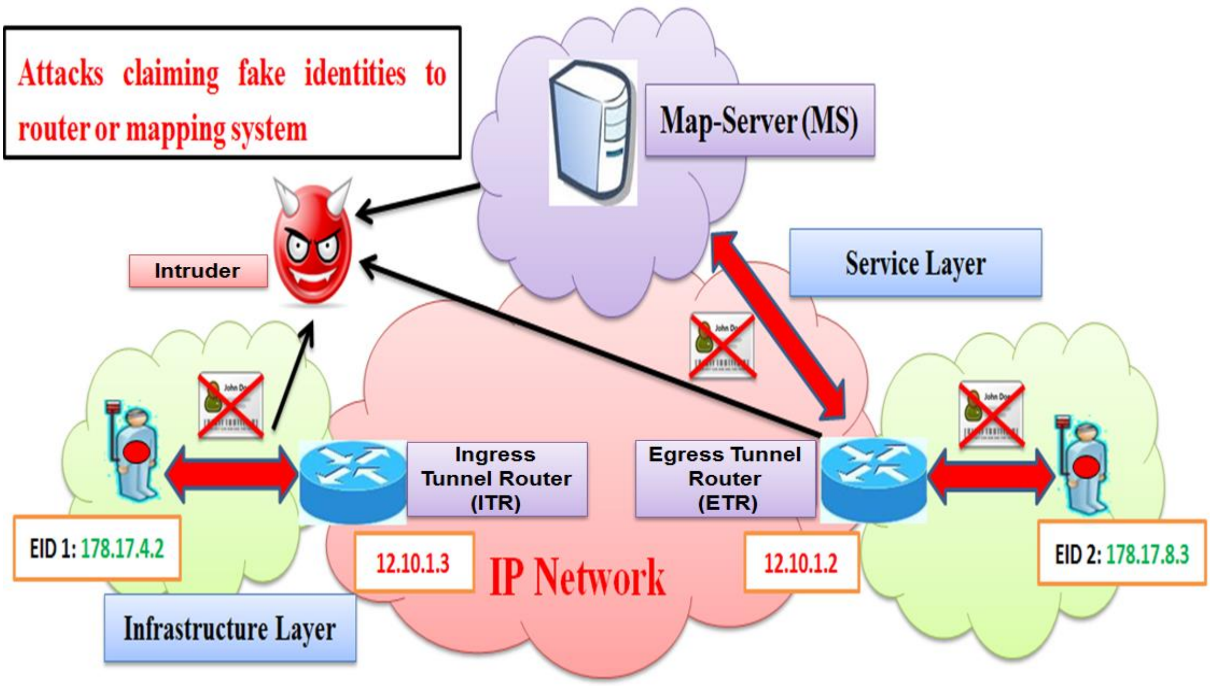


Fig 3.4 Threats in Authentication

3.4.3 Non-Repudiation

- **Infrastructure Layer:** Modules Control Plane & User Plane: The links (transactions) need to be secured between the XTR router and the Mapping system. For example, routers will not claim something they do not provide.

- **Service Layer:** Modules Control Plane & User Plane: Between the mapping system and the XTR, there is an insecure link. For example, by referring to the registration procedure after the router has registered, the device sends the queries to the mapping system in order to register and update this device. Through the insecure link, the mapping system can deny the acknowledgement to the router.

- **Threats and Attacks:** The attacker can act as either a fake router or a mapping system which cause disruption or interception of the data as shown in Figure 3.5. Here, the intruder can launch sinkhole attacks that make EID 1 or EID 2 devices believe that they are neighbour nodes and forward the packets between them. This cause confusion to the routers ITR and ETR and locator which the node receives make false data. Similarly the intruder can spoof on routers table of EID as well as on to the Map-Server table of Locator and EIDs.

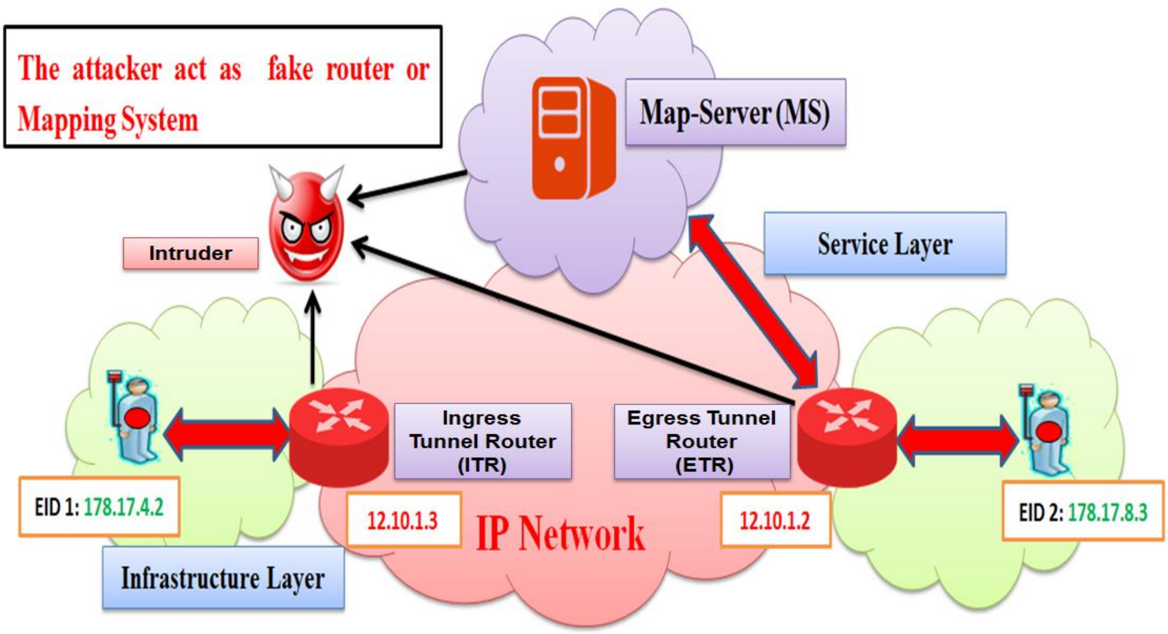


Fig 3.5 Threats in Non-Repudiation

3.4.4 Data Confidentiality

- **Infrastructure Layer:** Modules Control Plane & User Plane: There is no mechanism such as encryption/or encoding between the machine and the XTR router, and from machine to machine when they communicate with each other on different networks.

- **Service Layer:** Modules Control Plane & User Plane: There is no encryption mechanism between the (ITR and ETR) routers and between an XTR and a mapping system

- **Threats and Attacks:** As shown in Figure 3.6, two threats can occur here: EID Spoofing and RLOC spoofing. In EID spoofing, the originator of the packet puts in a spoofed EID and the packet will normally be encapsulated by the ITR of the site/or a PITR if the source site is not LISP enabled. As for the RLOC Spoofing, the originator of the packet directly generates an LISP-encapsulated packet with a spoofed source RLoC.

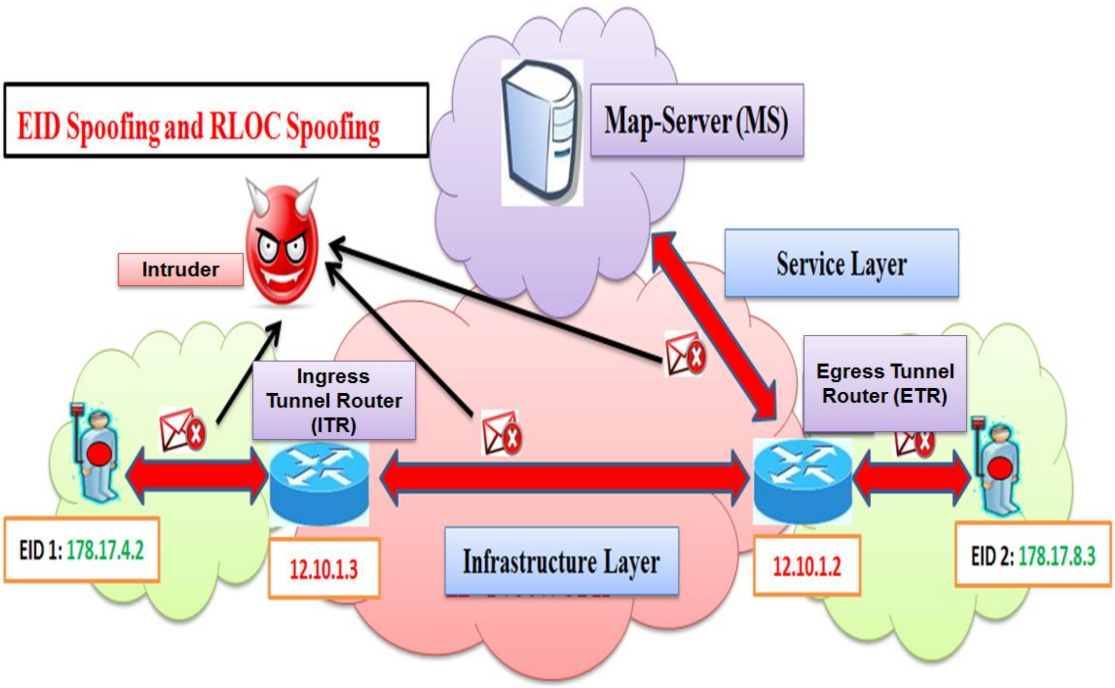


Fig 3.6 Threats in Data Confidentiality

4.4.5 Communication Security

- **Infrastructure Layer:** Modules Control Plane & User Plane: There is no security Tunnel/or End to End security between the machine and the XTR router, nor between the machine and other machines when they are communicating with each other on a different network.

- **Service Layer:** Modules Control Plane & User Plane: End to End is not exiting between the (ITR and ETR) routers and between the XTR and mapping system.

- **Threats and Attacks:** Eavesdropping/or spoofing can occur between the machine and the XTR router as shown in Figure 3.7. Furthermore, the attacker is able to capture and modify all the packets exchanged between an Ingress Tunnel Router (ITR) and Egress Router (ETR) and between the XTR and the mapping system.

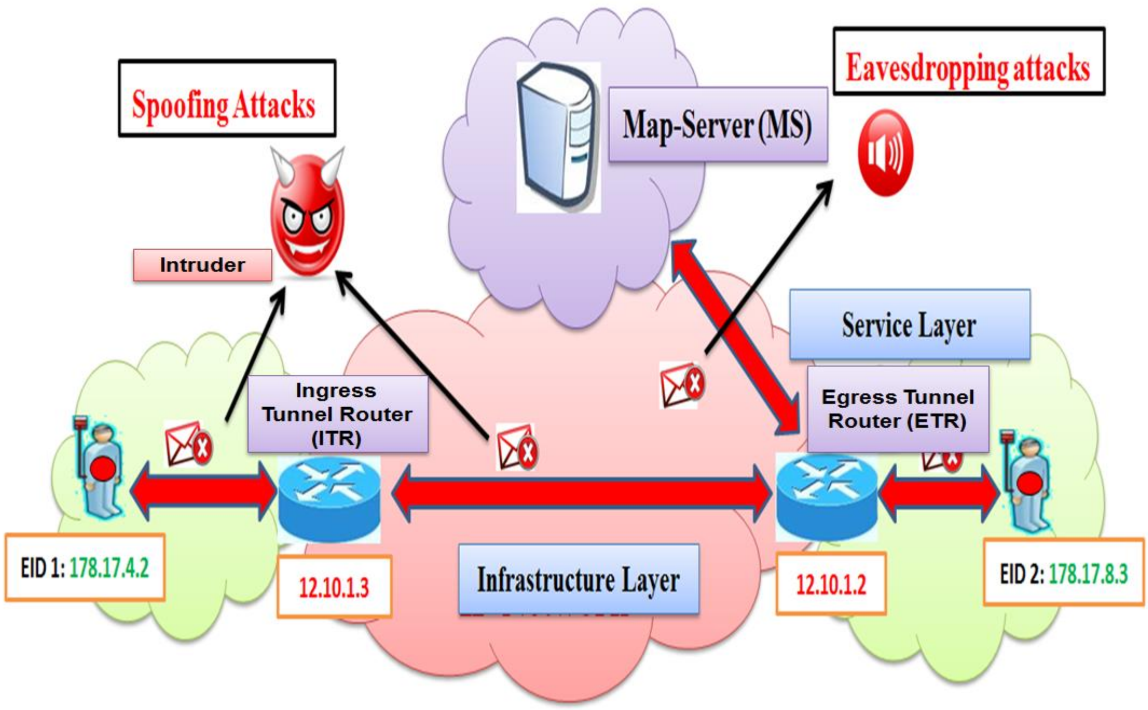


Figure 3.7 Threats in Communication

3.4.6 Data Integrity

- **Infrastructure Layer:** Modules Control Plane & User Plane: there are no mechanisms such as MD5/or digital signatures between the machine and the XTR router, and between the machine and other machines e.g. when machine (A) wants to communicate with machine (B) on a different network.

- **Service Layer:** Modules Control Plane & User Plane: there are No mechanisms such as MD5/or digital signature between Ingress and Egress and between the XTR and mapping system.

- **Threats and Attacks:** The intruder can capture the data between machine A and machine B, and establish a new spoofed connection. Adding to this, the intruder inserts itself in the middle of all connections, (i.e., Man in Middle Attack) as shown in Figure 3.8. Furthermore, the attacker can spoof on the transaction queries between (ITR and ETR) and the mapping system.

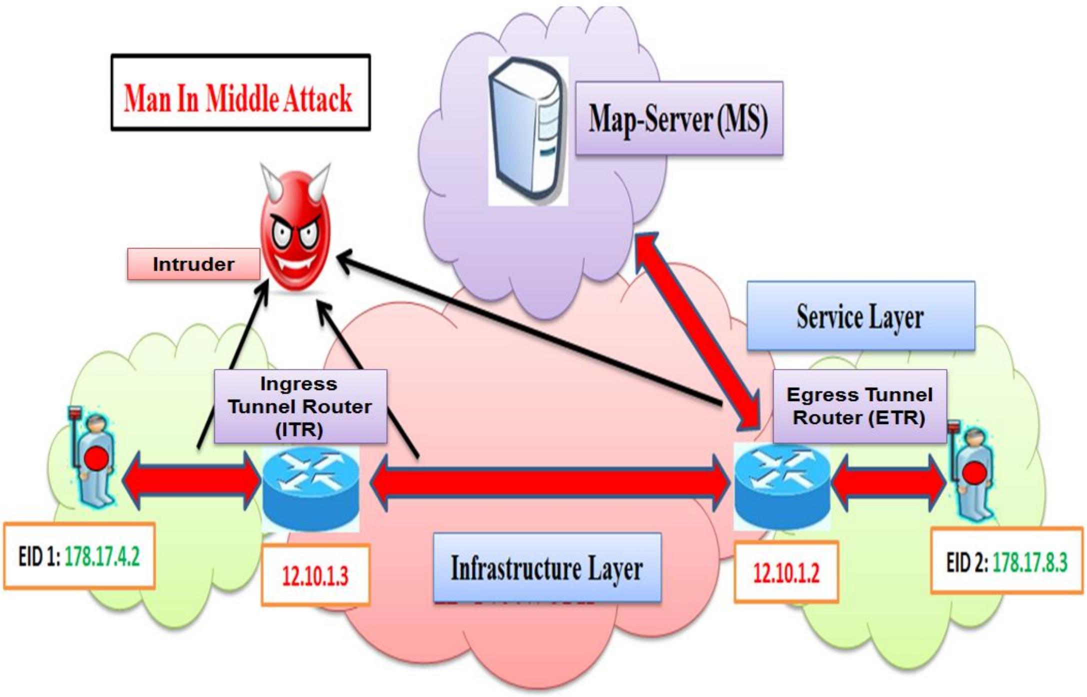


Fig 3.8 Threats in Data Integrity

3.4.7 Availability

- **Infrastructure Layer:** Modules Control Plane & User Plane: In the router, there is only one scenario with invalid information, i.e., Locator location and EID address

- **Service Layer:** Modules Control Plane & User Plane: Two kinds of Availability are needed: firstly, the Mapping system should not be overloaded and secondly, it should not have invalid information which affects the system scalability.

- **Threats and Attacks:** EID redirection/RLOC poisoning: The EID/or machine-Prefix in the mapping is not bound to (located by) the set of RLOCs present in the mapping. This could result in packets being redirected elsewhere, eavesdropped, or blackhole attack. Note: it is not necessary that RLOCs the highest priority ones are compromised. Moreover, DoS attacks can occur and intercept the packet between the machine and the XTR, between the machine and another machine on different networks and also between (ITR & ETR) and the XTR router and mapping system, as shown in Figure 3.9.

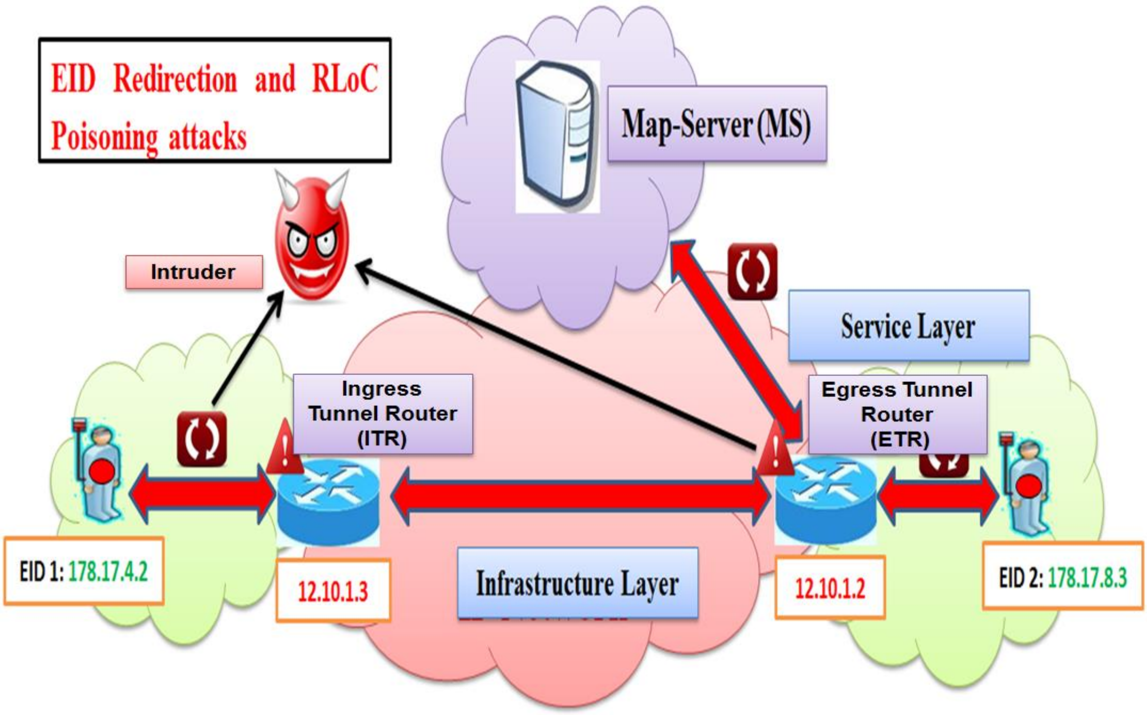


Fig 3.9 Threats in Availability

3.4.8 Privacy

- **Infrastructure Layer:** Modules 2 & 3: There is no privacy when data/or information is registering on the router leading to a security issue called information leaking.
- **Service Layer:** Modules 2 & 3: No privacy information is in the mapping system.
- **Threats and Attacks:** Unauthorized entity can disclose the privacy of the system.

In order to modell the security threats for the IoT based on LISP network architecture using AVISPA tool is used in order to design efficient protocols against these threats. Table 3.1 shows the headers of Automated Validation Internet Security Protocol and Applications

(AVISPA) the #basic roles; it is essentially needed to define the composed roles which describe the sessions of the protocol [AVISPA, 2013]. The #composed roles have no transition section, but rather a composition section in which the basic roles are instantiated. In the session role, it usually declares all the channels used by the basic roles. These variables are not instantiated with concrete constants. The channel type takes an additional attribute, in parentheses, which specifies the intruder model that assumed for that channel. The #intruder may play some roles as a legitimate user. There is also a statement which describes what knowledge the intruder initially has. Typically, this includes the names of all agents, all the symmetric keys and any shares with others. #Specifying Security Goals are specified in High level protocols specification language (HLPSL) by augmenting the transitions of the basic roles with the so-called goal facts and by then assigning them a meaning by describing, in the HLPSL goal section, what conditions – i.e. what combination of such facts indicate an attack and a violation of secrecy [AVISPA, 2013].

Table 3.1 The Headers of AVISPA Input File:

The Header	Description
# Basic roles	Define agents, variables and functions in the protocol.
# Transition	Represents the receipt of messages and the sending of reply messages, i.e., showing all the messages exchanges between the agents
# Composed role	It is one or more basic roles, 'gluing' them together so they execute together, usually in parallel with interleaving semantics.
# Session environment	Defines how agent parameters interact in the protocol.
# Intruder Information	Specifies the intruder's knowledge and capabilities
# Specifying Security Goal	Specifies the security properties to be checked.

3.5 Summary

This chapter has presented a comprehensive security analysis for the IoT network. In other words, it demonstrates the most common security issues and vulnerabilities in the IoT network. The analysis is based on the X.805 security standard and has, therefore, considered the security threats of IoT based on LISP network architecture. The analysis shows that there is a genuine need to provide new mechanisms to enforce Access control, Authentication, Non repudiation, Data confidentiality, Communication Security, Data Integrity, Availability and Privacy. The chapter, then, has to consider two layer threats namely, the infrastructure and Service layers to expose the security threats during the unsecure signal transactions between the IoT, the LISP-capable routers and the mapping system. As a result, a number of security vulnerabilities have been discovered and described. Therefore, table 3.2 shows the comparison summarising the analysis of security issues

Table 3.2 Comparison security Issues in IoT

Access Control:	
Infrastructure Layer:	Modules 2 & 3, no Packet filtering between ITR and ETR
Service Layer:	Modules 2 & 3,no Packet filtering between ITR and ETR
Threats and Attacks:	Unauthorized devices/ or illegitimate devices access to the network and the Dos attacks.
Authentication:	
Infrastructure Layer:	Modules 2 & 3, users/machines that register themselves with XTR router require authentication to ensure that whoever the user or machine claims to be.
Service Layer:	Modules 2 & 3, The XTR is registered with the mapping system, it needs to be authenticated with another router.
Threats and Attacks:	The attackers (Unauthenticated devices) can access the network by claiming fake identities to the router or the mapping system
Non-Repudiation:	
Infrastructure Layer:	Modules 2 & 3, the links (transactions) need to be secured between XTR router and Mapping System
Service Layer:	Modules 2 & 3, between the mapping system and the XTR there is an insecure link
Threats and Attacks:	The attacker can act as either a fake router or a mapping system which causes disruption or intercepts the data.
Data Confidentiality:	
Infrastructure Layer:	Modules 2 & 3,no mechanism such as encryption/ or encoding between the machine and the XTR, and from machine to machine when they communication with each other on different networks
Service Layer:	Modules 2 & 3, There is no encryption mechanism between the XTR and a mapping system.
Threats and Attacks:	Two threats : EID Spoofing and RLoC Spoofing
Communication Security:	
Infrastructure Layer:	Modules 2 & 3, there is no security Tunnel/ or End to End security between the machine and the XTR
Service Layer:	Modules 2 & 3, end to End is not exiting between the (ITR & ETR) router and between the XTR and mapping system.
Threats and Attacks:	Eavesdropping/ or spoofing can occur between the machine and the XTR router.
Data Integrity:	
Infrastructure Layer:	Modules 2 & 3, no mechanisms such as MD5/s or digital signatures between the machine and the XTR router.
Service Layer:	Modules 2 & 3, no mechanisms such as MD5/ or digital signature between Ingress and Egress and between the XTR and mapping system.
Threats and Attacks:	The intruder can capture the data between machines A and machine B, and establish a new spoofed connection. Also, the intruder inserts a new spoofed connection.
Availability:	
Infrastructure Layer:	Modules 2 & 3, In the router, there is only one scenario with invalid information i.e. Locator location and EID address.

Service Layer:	Modules 2 & 3, Two kinds of Availability are needed: Mapping system should not be overloaded and do not have invalid information which affects the system scalability.
Threats and Attacks:	EID redirection/ RLoC poisoning The EID/or machine-Prefix in the mapping is not bound to located by the set of RLoCs present in the mapping. This could result in packets being redirected elsewhere, eavesdropped or black holed
Privacy:	
Infrastructure Layer:	Modules 2 & 3, There is no privacy when data/ or information is registering on the router
Service Layer:	Modules 2 & 3, Unauthorized entity can disclose the privacy of the system.
Threats and Attacks:	Unauthorized entity can disclose the privacy of the system.

Based on all these analyse, a strong End-to-End security authentication protocol is basically needed to the IoT devices. Accordingly, chapter 4 will discuss the registration security protocol, its design and verification in IoT based on LISP network architecture.

Chapter 4

An Enhanced Security Protocol for Registration Stage in LISP Network Architecture

4.1 Introduction

The Locator/ID Separation Protocol (LISP) is a routing architecture that provides new semantics for IP addressing. To simplify routing operations and improve scalability in Internet of Things (IoT), The LISP separates the device identity from its location using two different naming spaces. As mentioned earlier in chapter 2, LISP introduced a mapping system match to two spaces. In the initial stage, each LISP-capable router needs to register with a Map Server known as the Registration Stage. Nevertheless, this stage is vulnerable to impersonating and content poisoning attacks. Consequently, this chapter introduces an enhanced security protocol in Registration stage using ID/Based Cryptography (IBC) method. The security protocols have been verified using formal methods approach via Automated Validation Internet Security Protocol and Applications (AVISPA) tool. Hence, the structure of this chapter is divided into two important sections as the following: section 4.2 discusses the first version of designed enhanced security protocol and various discovered attacks found by (AVISPA) tool. Whereas, section 4.3 discusses the final version of the enhanced security protocol using IBC method and is verified by AVISPA tool. Finally, a summary concludes the chapter in section 4.4.

4.2 First Version Enhanced Security Protocol for Registration Stage in LISP Architecture

As mentioned in chapter 4, LISP suffers from different security vulnerabilities; therefore, a new security method for protecting the LISP stage is presented. The proposed protocol provides a Two-party mutual authentication, by using a trust authentication server (TAS) as a third party trusted between ETR and MS.

4.2.1 Protocol Exchange Messages:

Step 1A: ETR → TAS: (ETR, N1, Map- Request Register), {h (N1, Map-Register} {MS}

Here, in step1A, ETR sends a Map request Register to TAS including a 4-byte random nonce (N1), in order to request a Map-Register from MS server. The ETR expects to receive the same nonce in Map-Reply message from the TAS.

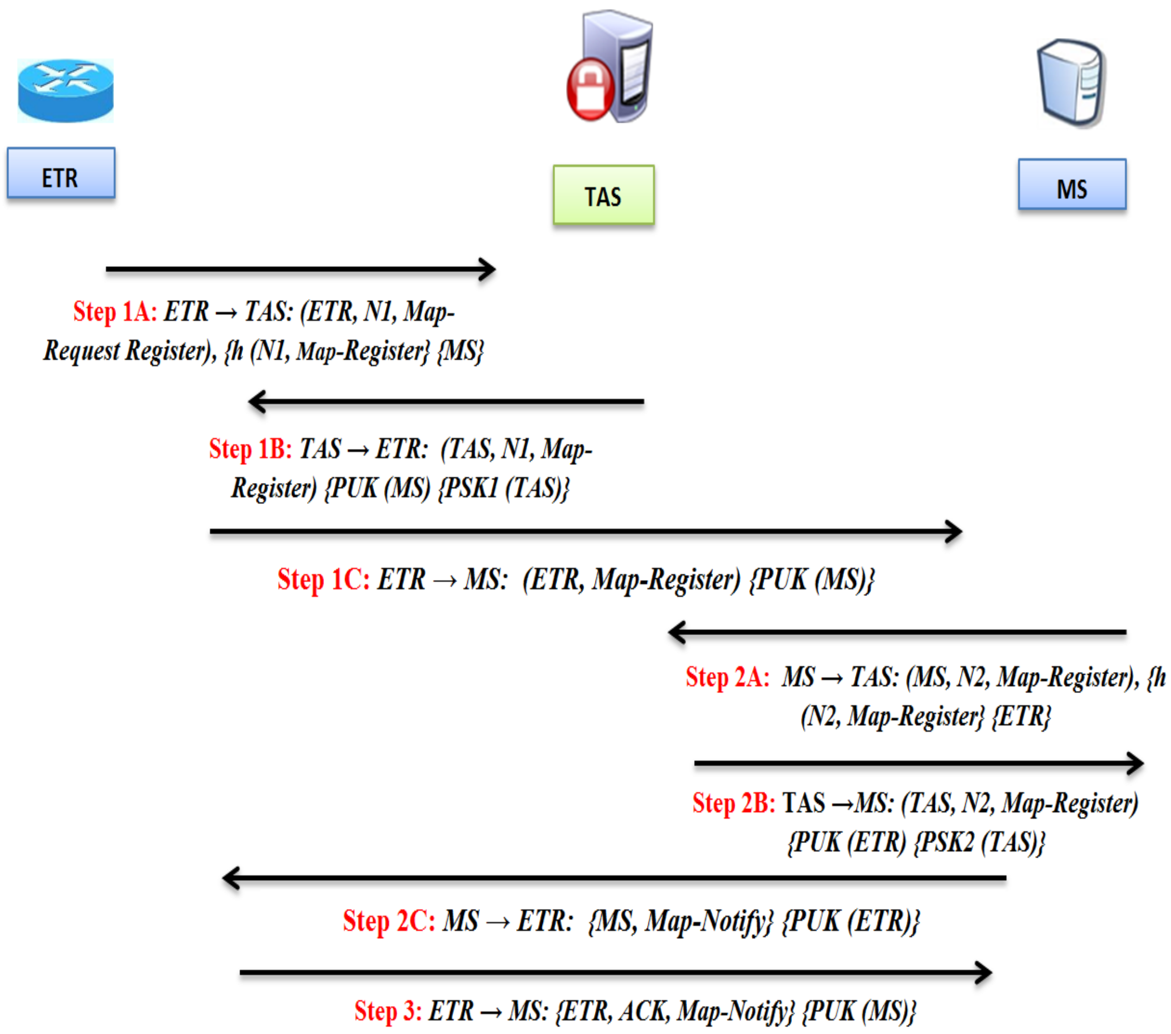


Fig 4.1 Security Protocol for Address Registration

Step 1B: TAS → ETR: (TAS, N1, Map-Register) {PUK (MS) {PSK1 (TAS)}

Step 1B, the TAS replies to the ETR and provides the public key of MS encrypted with PSK1 of TAS.

Step 1C: ETR → MS: (ETR, Map-Register) {PUK (MS)}

Step 1C, ETR sends LISP Map-Register Packet to MS. This message is encrypted by MS public key and the MS decrypts the message using its private key.

Step 2A: MS → TAS: (MS, N2, Map-Register), {h (N2, Map-Register} {ETR}

Step 2A, the MS forwards the message to the TAS that is received in Step 1C; this message includes random nonce (N2) to register the new device of ETR request. The MS expects to receive the same nonce in Map-Reply message from the TAS.

Step 2B: TAS →MS: (TAS, N2, Map-Register) {PUK (ETR) {PSK2 (TAS)}

Step 2B, the TAS replies to MS and provides the public key of ETR encrypted with PSK2 of TAS.

Step 2C: MS → ETR: {MS, Map-Notify} {PUK (ETR)}

Step 2C, the MS registers the device and sends a Map-Notify message to ETR encrypted with the Public key of ETR router.

Step 3: ETR → MS: {ETR, ACK, Map-Notify} {PUK (MS)}

Upon Step 2C, the ETR decrypts the message using its private key. The ETR then sends in Step 3 acknowledgment to the MS to confirm that Map-Notify has been received and updated its router table. This message is encrypted using the Public key of MS.

Table 4.1 Notation of Register Stage in LISP Network Architecture

The Notation	Definition
TAS	Trust Authentication Server
PUK (ETR) , PUK(MS)	The Public Keys of the ETR and the MS, respectively. These keys are derived by the TAS
PSK1,PSK2	Pre-shared keys to secure the connections between the TAS and ETR, MS
ETR	The Egress Tunnel Router in the destination EID Space
MS	The Map Server
N1,N2	A fresh random number
h(m)	Hash value of the message (m)

4.2.2 Formal Analysis and Attacks Discovered Using AVISPA

The first version of security registration protocol for LISP network architecture is simulated using AVISPA tool. The AVISPA file describes the system Headers via table 3.1 which has been introduced earlier in chapter 3. In brief, only the #Specifying security Goals and the #Intruder Information heading are described here while the rest is less significant in terms of understanding the verification process. The security requirements of the system are defined under the # specifying Security Goals. The PUK_ETR and PUK_MS are the public keys which are provided by Trust Authentication Server TAS, these keys are encrypted by PSK1 and PSK2 Pre-Sheared key that secures the connection between TAS and ETR, and between TAS and MS. The n1 and n2 random numbers (nonce) are used as challenge response between the ETR and the TAS on one hand, and between the MS and the TAS as secure request response on the other. Table 4.2 shows It has been set in this protocol as Weak_authentication on etr_tas, and Weak_authentication on ms_tas. The Weak authentication (X, Y) specifying goal means that if Y thinks he has successfully completed a run of the protocol with X, then X has previously been running the protocol with Y. The authentication_on router_map_server_n1 and authentication_on map_server_router_n2 is set as strong authentication between TAS and ETR and TAS and MS.

Table 4.2 HLPSL Code: Specifying Security Goal

HLPSL Code in AVISPA	**Comments**
Goal	% Specifies the security properties to be cheeked
secrecy_of sn1, sn2,etr, ms, tas	% Check the secrecy of random nonce N1 and N2 between the MS and TAS
Weak_ authentication on etr_tas	% Check the authentication between ETR and TAS, and then ETR should provide confirmation witness information to TAS.
Weak_authentication on ms_tas	% Check the authentication between MS and TAS, and the MS should provide a confirmation witness information to TAS
authentication_on router_map_server_n1	% strong authentication between router and map server on value of random nonce N1.
authentication_on map_server_router_n2	% strong authentication between map server and router on value of random nonce N2.
End goal	%End of session goal

Table 4.3 shows, the #Intruder Information heading specifies the intruder identity, knowledge and capability. The first line identifies the intruder knowledge and defines the Intruder's initial Knowledge, i.e., it has been assumed that intruder knows the identity of the participants and can fabricate the map-register and map-notify by intercepting the connection and capturing information.

Table 4.3 HLPSL Code: Intruder Information Heading

HLPSL Code in AVISPA	**Comments**
intruder_knowledge = {etr,ms,psk1_tas, psk2_tas, puk_etr, puk_ms, puki,pski, inv(ki)}	% Specifies the intruder's knowledge and capabilities. Check the intruder between the ETR and TAS and between MS and TAS on pre-shared keys PSK1 and PSK2. And also check if intruder captures the public keys PUK between the ETR and MS.
composition	% It is one or more basic roles, gluing them together so they execute together, usually in parallel with interleaving semantics.
map_server(ms,etr,psk1_tas, psk2_tas KeyMap, Snd, Rcv)	%set the pre-shared keys PSK1 & PSK2 between the MS and TAS and between the ETR and MS.
)/\ nspk(Snd, Rcv,	%channels
Psk1, psk2	% pre-shared keys to secure the connections between the TAS and ETR , MS.
{etr.ms.puk_etr.puk_ms,etr.i.puk_etr.ki,i.m	% check the intruder between the ETR and MS on public keys

s.ki.puk_ms}, {etr.{etr.puk_etr,ms.puk_ms}, KeyRings ms.{ms.puk_ms},i.{i.ki}})	PUK_ETR and PUK_MS
end role	%End of session role

4.2.3 Security Analysis for the Registration Protocol:

After generating the HLPSL description of the system using the AVISPA tool to check the security assertion, the following attacks have been discovered by AVISPA tool:

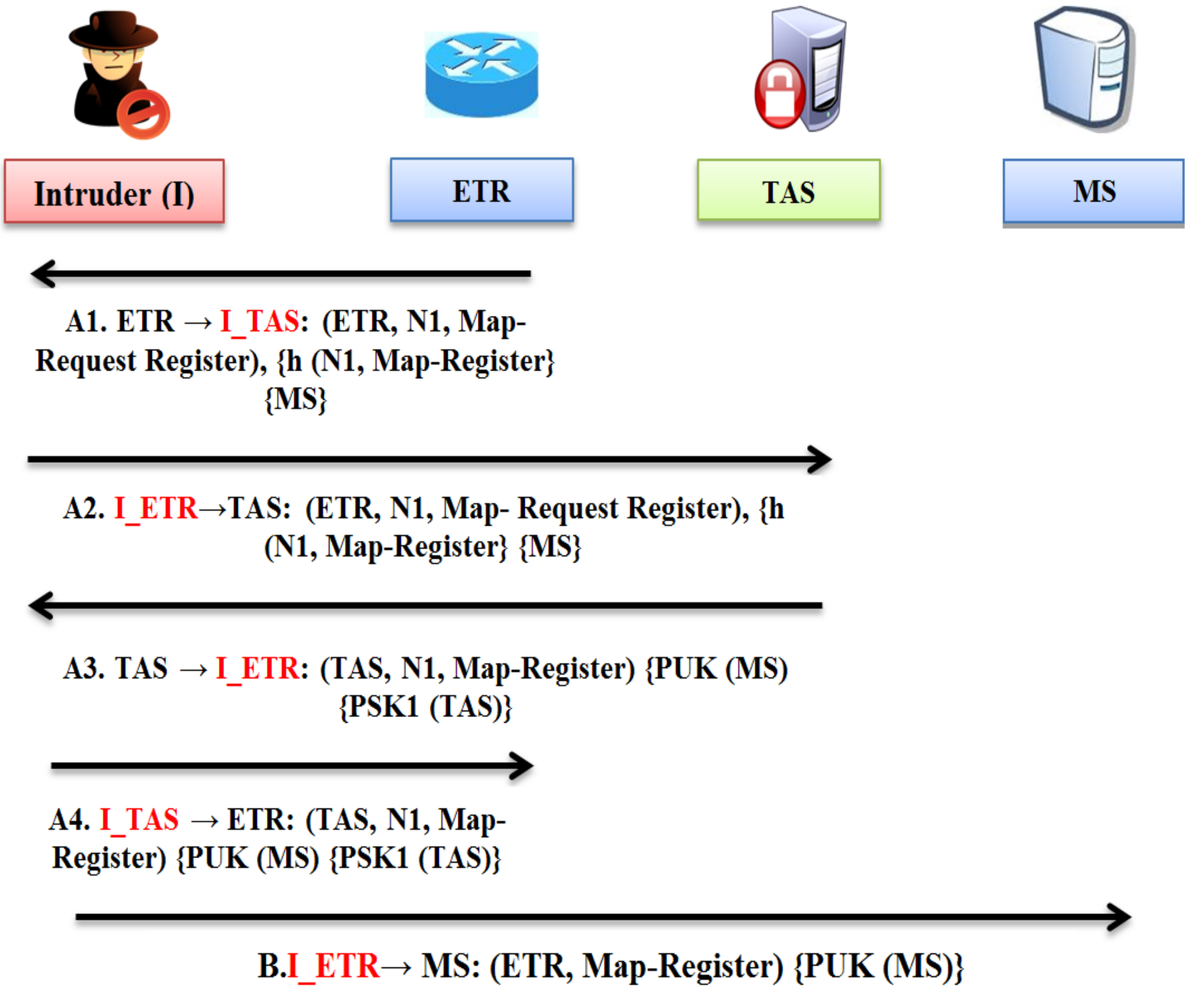


Fig 4.2 Attack 1: on I_ETR and I_TAS Playback Attacks

Figure 4.2 shows Attack1, the intruder intercepts the messages (A1) to (B1) and replays as the ETR and TAS by acting both of the I_ETR and I_TAS. This is called playback attack, where the Intruder (I) intercepts the data and retransmits or redirects it to its own direction. Since, there is no encryption, the intruder acquires the N and uses it to impersonate the ETR.

Consequently, the TAS runs this process believing that it is communicating with ETR while in reality it is communicating with the Intruder.

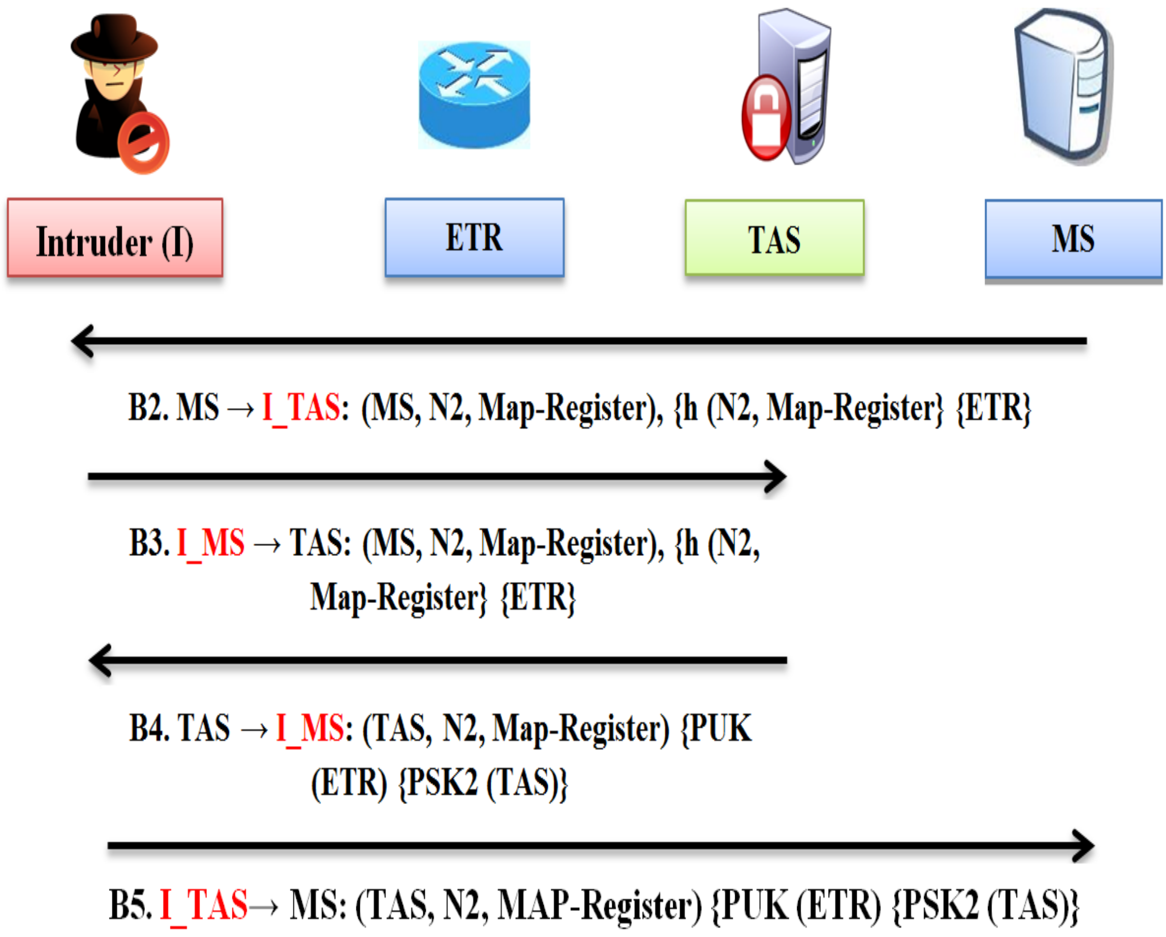


Fig 4.3 Attack 2: on I_MS and I_TAS Eavesdropping Attacks

Figure 4.3 shows Attack2, the MS sends message (B2) to prove its identity and request PUK of ETR from TAS in order to communicate with ETR. This message is transformed like a hash function (h), meanwhile, the Intruder (I) eavesdrops on the conversation and keeps hash. When the interchange is over, the intruder is as either MS or TAS, when TAS asks for a proof of identity, the intruder sends ETR hash read from the first session, which TAS accepts and thus grants access to the intruder. This attack is called eavesdropping attack.

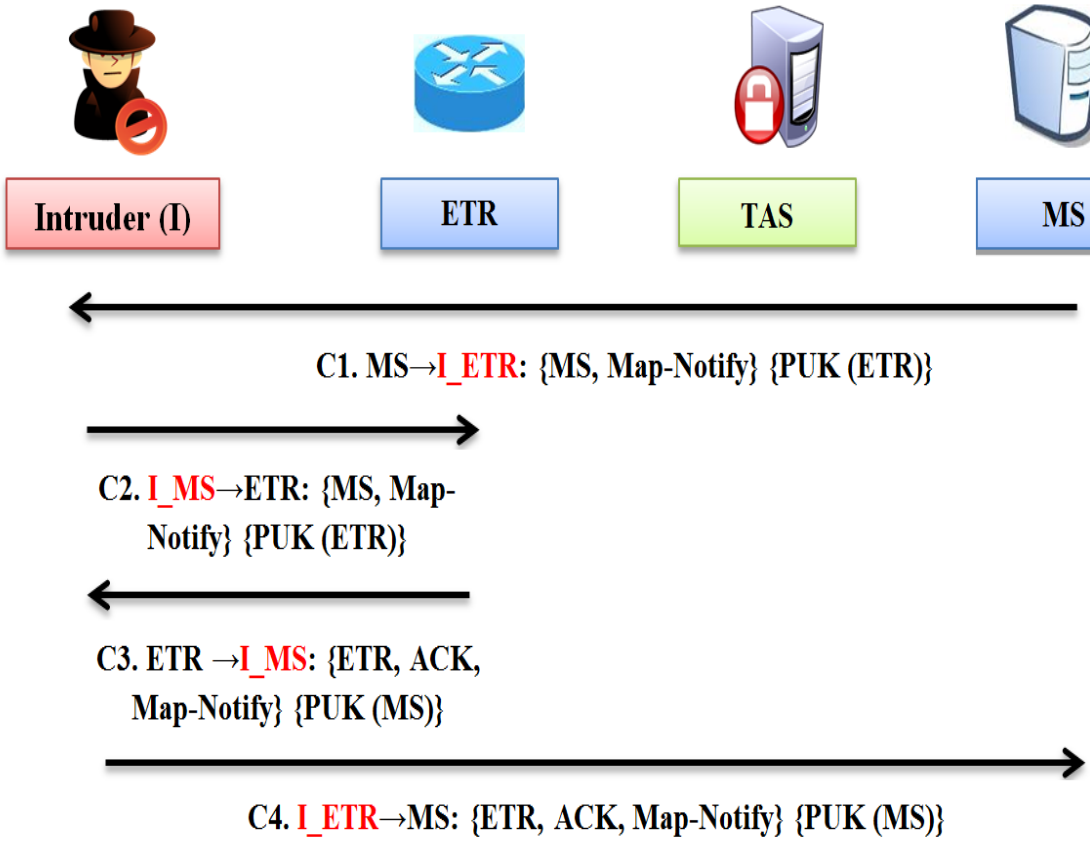


Fig 4.4 Attack 3: on I_ETR and I_MS Man in the Middle Attacks

#Attack3, Here, the intruder intercepts the communication and acts as I_ETR, I_MS between the ETR and MS. It impersonates (imitates or mimics) both parties and gains access to information when the two parties are trying to send messages to each other. This attack is called MitM, which allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late, as messages (C1) to (C4) show in figure 4.4.

4.3 Final Version Security Protocol Enhancement for Registration Stage in LISP Network Architecture

As it has been mentioned previously in above, the first version of the designed protocol has security vulnerabilities such as playback attack, eavesdropping attack and MitM; therefore, an enhanced security protocol for the registration stage is presented in this section. This security protocol method uses the ID-Based cryptography (IBC) which allows the mapping system to authenticate the data.

4.3.1 Protocol Exchange Message

The registration procedure using the IBC goes as follows:

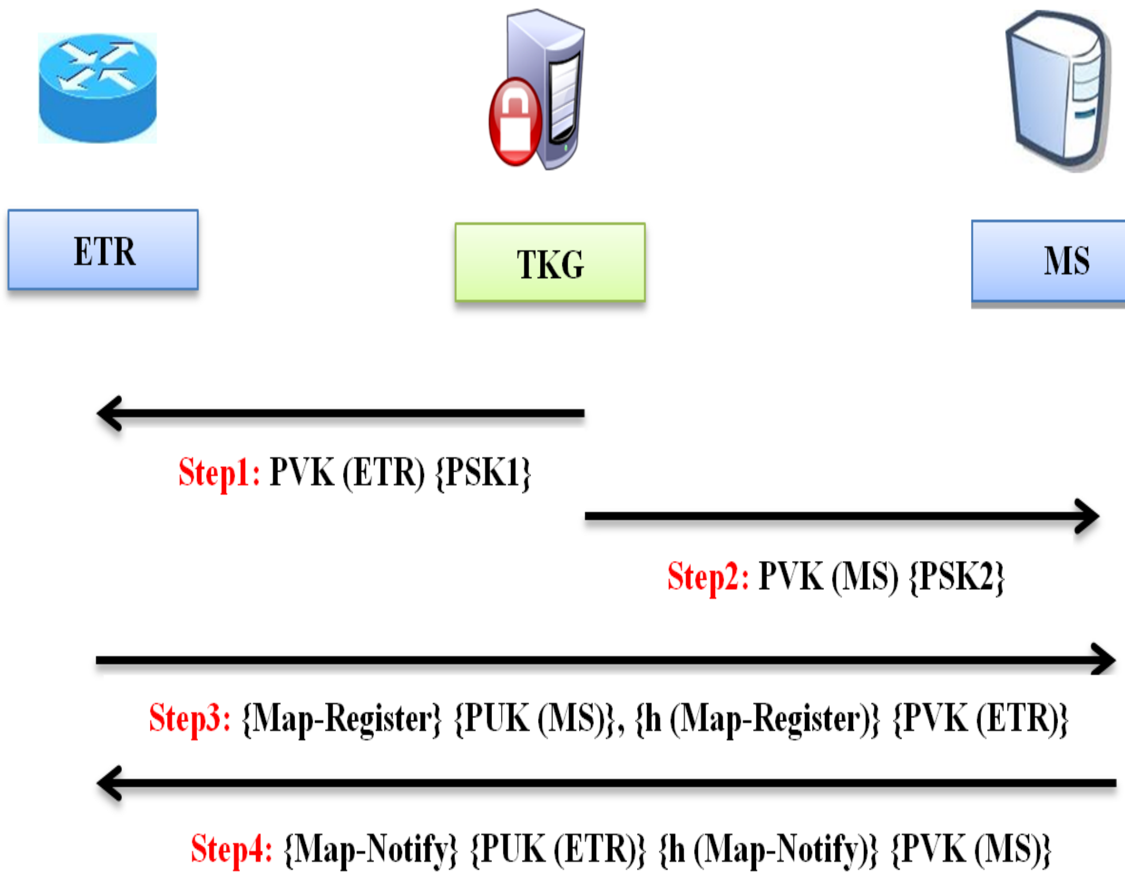


Fig 4.5 Security Protocol for Address Registration

Step 1: $\mathbf{TKG \rightarrow ETR} : \{\ \mathbf{PVK(ETR)\{PSK1\}}$

Step 2: $\mathbf{TKG \rightarrow MS} : \{\ \mathbf{PVK(MS)\{PSK2\}}$

The TKG provides the two communicating parties (ETR, MS) with their private keys PVK (ETR) and PVK (MS) in steps 1 and 2. These two steps are encrypted using the pre-shared secret keys PSK1 and PSK2 respectively.

Step 3: $\mathbf{ETR \rightarrow MS}: \{\mathbf{Map - Register}\}\{\mathbf{PUK(MS)}\}, \{\mathbf{h(Map - Register)}\}\{\mathbf{PVK(ETR)}\}$

The ETR sends a LISP Map-Register packet in step 3. The content of this message is encrypted using the MS's public key which is known publicly and signed digitally using the private key of the ETR. As described in [Cisco-A, 2014], the Map-Register packet includes the ETR's address (RLOC), a random number (n1) and a list of EID-Prefix managed by ETR.

Step 4: $\mathbf{Ms \rightarrow ETR: \{Map - Notify\}\{PUK(ETR)\}, \{h(Map - Notify)\}\{PVK(MS)\}}$

Upon receiving Step 3, the MS will use its private key PVK (MS) to decrypt the message and then verify the signature using the ETR's public key PUK (ETR). Finally, the MS will hash the included Map-Register and compare the result with the received signed value. Only if the two values are equal, the MS can compose a Map-Notify packet as Step 4 which includes the received random number (n1). This message is encrypted using the ETR's public key and signed digitally using the MS's private key. The ETR will check the included random number and only when the check succeeds, the ETR authenticates the MS.

Table 4.4 Notation of Registration Stage in LISP Network Architecture

The Notation	Definition
TKG	The Trusted Ticket Granting
PVK (ETR) , PVK(MS)	The Private Keys of the ETR and MS respectively. These keys are derived by the TKG
PUK	Public key
PSK1,PSK2	Pre-shared keys to secure the connections between the TKG ,ETR and MS
ETR	The Egress Tunnel Router in the destination EID Space
MS	The Map Server
n1	A fresh random number
h(m)	Hash value of the message (m)
{m}{K}	The message (m) being encrypted with the key (K)

4.3.2 Formal Analysis Using AVISPA

To formally analyse the basic mapping procedure, the system will be simulated via AVISPA tool. The full HLPSL input file describing the system is included in Appendix-D. For conciseness, only #Transitions, the #specifying security Goals and the #Intruder Information heading are described here while the rest is less significant in terms of understanding the verification process.

In table 4.5 shows the #Transitions heading defines the system and the transitions between the entities. It is worth pointing out that for security simulation, it is essentially needed to define explicitly the security parameters. Therefore, the security related contents of the Map-Register and Map-Notify have been shown in step 3.

Table 4.5 HLPSL Code: Transitions

HLPSL Code in AVISPA	Comments
Transition	% Represents the receipt of messages and the sending of reply messages, i.e. showing all the messages exchanges between the agents
1. State = 0 /\ Rcv(start) =|> State' := 1 /\Snd(TKG.ETR.{PVK_ETR}_PSK1	% The ETR received the private key of PVK_ETR from the TKG and encrypted by Pre-Shared Key PSK1.
2. State = 1 /\ Rcv(Start) =|>State' := 2 /\ Snd(TKG.MS.{PVK_MS}_PSK2	% The MS received the private key of PVK_MS from the TKG and encrypted by Pre-Shared Key PSK2.
3.State = 2 /\ Rcv(PVK_ETR)PSK1) =|> State' := 3 /\Snd(ETR.MS.{M,ETR,N1,EIDPre}{PUK_MS}, {H(M,ETR,N1,EIDPre)}{PVK_ETR} /\witness(ETR,MS,N1)	% The ETR sends a Map-Register pack in State 3. The content of this message is encrypted using MS public key PUK which is known publicly and signed digitally using the private key of the ETR. The Map-Register packet includes the ETR address, random number (N1) and a list EID-Prefix managed by ETR.
4.State = 3 /\ Rcv(PVK_MS)PSK2) =|>State' := 4 /\Snd(MS.ETR.{m2,N1}{PUK_ETR},{H(m2,n1)}{PVK_MS)} /\ witness(MS,ETR,N1) /\ secret (PUK_ETR,PVK_MS,{MS,ETR})	% In the State 4, the MS will use its private key PVK_MS to decrypt the message and then verify the signature using the ETR public key PUK_ETR. Then MS will hash the included Map Register and compare the result with the received signed value.
End role	%End of session transition

The security requirements of the system are defined under the table 4.6 shows the #Specification Security Goals heading. The lines starting with the keyword Secret define the secrecy properties of the protocol. secret_of Map_server and Router_ETR on n1 specifies the n1 nonce as a secret between the ETR and the MS. The line starting the ROUTER_ETR authenticates Server_MS on n1. Authentication on_n1 defines the protocol's authenticity properties, where the MS is authenticated correctly to ETR using the random number n1. The weak_authentication ETR and MS assertion could be explained as follows: if ETR has completed a run of protocol with MS, then MS has previously been running the protocol, apparently with ETR.

Table 4.6 HLPSL Code: Specification Security Goals

HLPSL Code in AVISPA	Comments
Goal	% Specifies the security properties to be cheeked
security_of secEtrMS, secMsEtr	% Check the secrecy of between ETR and MS and between MS and ETR in order to avoid any intruder between these two entities.
weak_authentication_on _ETR	% Check the authentication of ETR, and then ETR should provide confirmation witness information to MS.
weak_authentication_on_MS	% Check the authentication of MS, and the MS should provide confirmation witness information to ETR.
authentication on_n1	% strong authentication between router ETR and map server MS on value of random nonce N1.
End goal	%End of session goal

Table 4.7 shows the # Intruder Information heading specifies the intruder's identity, knowledge and capability. The first line identifies the intruder; the intruder knowledge defines the Intruder's initial knowledge. In other words, it has been assumed that the intruder knows the identity of the participants, their own private key, and can fabricate Map-Register and Map-Notify messages.

Table 4.7 HLPSL Code: Intruder Information Heading.

HLPSL Code in AVISPA	**Comments**
intruder_knowledge = {ETR,MS,ETR,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,H,M,M2}	% Specifies the intruder's knowledge and capabilities. Check the intruder between the ETR and MS on pre-shared keys PSK1 and PSK2. And also check if intruder captures the public keys PUK between the ETR and MS.
composition	% It is one or more basic roles, gluing them together so they execute together, usually in parallel with interleaving semantics.
session(etr,ms,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,h) /	%set keys PSK1, PSK2, PUK and PVK of ETR and MS
session(etr,i,PVK_ETR,h)/	%Check the intruder between the ETR and the received PVK ETR from the TKG.
session(Ms,i,PVK_MS,h)	%Check the intruder between the MS and the received PVK MS from the TKG.
end role	%End of session role

4.3.3 Analysis of Results

Four tools in Automated Validation of Internet Security Protocols and Applications (AVISPAP) have been used in order to check and verify any security vulnerabilities in the registration protocol. Figure 4.6 shows; the current version of the tool integrates four back-ends: the On-the-fly Model-Checker OFMC, the Constraint-Logic-based Attack Searcher CL-AtSe, the SAT-based Model-Checker SATMC and the TA4SP protocol analyser, they verifies protocols by implementing tree automata based on automatic approximations. All the back-ends of the tool analyse protocols under the assumptions of perfect cryptography and that the protocol messages are exchanged over a network that is under the control of a Dolev-Yao intruder [AVISPA, 2013]. That is, the back-ends analyse protocols by considering the standard protocol independent, asynchronous model of an active intruder who controls the network but cannot break cryptography; in particular, the intruder can intercept messages and analyze them if he possesses the corresponding keys for decryption, Furthermore he can generate messages from his knowledge and send them under any party name.

Fig 4.6 AVISPA Results

Table 4.8 shows the results of the proposed security protocol for the registration stage in LISP architecture, while the input code and results are described in the appendix section -D. Furthermore, AVISPA is used to verify the security properties like secrecy, integrity and authentication. It gives details about whether the protocol is safe or not. If not then it also gives the trace of the attacks found, to indicate secrecy attack or authentication attack. So even though many properties of the protocol are to be checked, but only few can be verified using AVISPA. For this reason, the modelling is done in HLPSL language used by AVISPA tool. For our verification, OFMC, ATSE, SATMC, and TA4SP have been used to search for the attacks on the protocol. The feature of finding and tracing the attack makes AVISPA different from other tools. Hence the tool is tested and the protocol verification results are analysed; the results show that they do not have any security flaws and the security protocol of registration stage is safe to be used.

Table 4.8 AVISPA Tools (OFMC, ATSE, SATMC, and TA4SP) Results

Version	Tool	Description	Result
Basic session	OFMC	Visited Nodes: 376 nodes Depth: 11 plies Search Time: 0.9s	**SAFE**
Basic session	ATSE	Analysed : 23states Reachable : 6 states Translation: 0.00 seconds Computation: 0.05 seconds	**SAFE**
Basic session	SATMC	Attack Found false Boolean upper Bound Reached true boolean graphLeveledOff 4 steps satSolver zchaff solver maxStepsNumber 11 steps stepsNumber 5 steps atomsNumber 542 atoms clausesNumber 1716 clause e ncodingTime 0.36 seconds solvingTime 0.006 seconds if2sateCompilationTime 0.09 seconds ATTACK TRACE %% no attacks have been found...	**SAFE & goal as specified**
Basic session	TA4SP	STATISTICS SECURITY-As specified ATTACK TRACE No attack found	**SAFE**

4.3.4 Security Protocol Analysis for Registration in LISP Network Architecture

In spite of the fact that no attack has been discovered against the proposed solution in section 4.3.1, this result needs to be considered carefully. The formal verification result is based on the system defined in chapter 2 section 2.13. In this system, it has been assumed that the ETR knows the authoritative MS in its network or domain which is a very similar way to the current Domain Naming System (DNS), where clients are preconfigured with the authoritative DNS server. However, it simulates the case when the ETR is not sure of the identity of its authoritative MS. The following attack against the Secret_of Server_MS, Router_ETR on n1, ROUTER_ETR authenticates Server_MS on n1 and weak_ authentication Router_ETR, Server_MS assertions where discovered. Here the notations I_MS, I_ETR and I_TKG represent the case where the Intruder impersonates the MS, ETR and TKG, respectively. This is an active MitM; the Intruder intercepts and replays message A1 and A2 as shown in figure 4.7 this because the packets are sent in clear text. Since the ETR is not sure of the identity of the MS, the intruder manages to impersonate the MS and fools the ETR to use its public key rather than the MS's public key to encrypt message A3.

Consequently, the random number (n1) will be compromised and ETR will run the protocol mistakenly believing it to be with the MS, while in reality it is with the Intruder in B1, B2 and B3

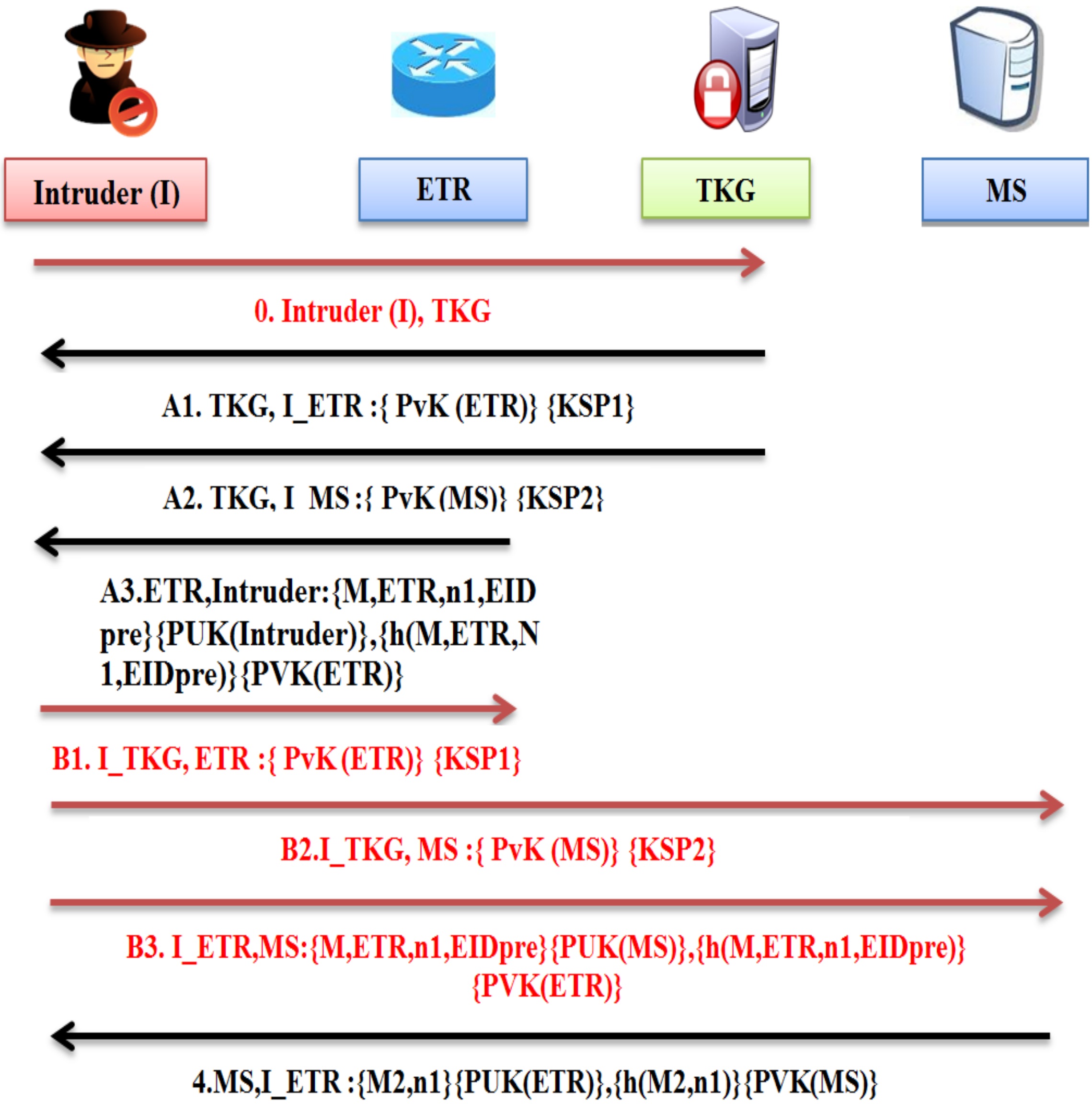


Fig 4.7 Attacks Discovery on Registration

In order to stop such attacks, the ETRs should be configured to use the authoritative Map Server in its domain or network. This could be achieved simply during the network configuration in a similar way to configure the default DNS server or the default Gateway in a network. Since the TKG enables users to communicate security and verify each other's signature without exchanging public or private keys, the protocol prevents Dos attacks on the network.

4.4 Summary

In this chapter, an enhanced security protocol had been introduced to secure the registration stage in the LISP architecture. The first designed version enhanced protocol used a Trust Authentication Server (TAS) as a third party trusted between ETR and MS, in order to provide two-party mutual authentication to entities i.e. ETR and MS. Noticeably, attacks had been discovered by AVISPA; these attacks were divided into three categories in the registration protocol. (I) attacks on I_ETR and I_TAS; here the intruder intercepts the messages between ETR and TAS: These attacks are called playback attacks. (II) attacks on the I_MS and I_TAS, here the intruder listens to the conversation between MS and TAS which causes disclose of the confidentiality information of these devices, i.e., MS and TAS. These attacks are called eavesdropping attack. (III) The third category is attack on I_ETR and I_MS where the intruder intercepts the communication and acts as I_ETR or I_MS conversation between the ETR and MS. These are known as Man in Middle Attacks (MitM). Therefore, the final version of enhanced security protocol for registration stage in LISP architecture provided a new security method based on the IBC, allowing a Map-Server to check the received information (i.e. the EID-Prefix) and providing secure authentication as well. The protocol is verified by AVISPA tool and the results show that there are no security flaws.

As chapter 4 has introduced an enhanced security protocol for the Registration Stage, Chapter 5 introduces an enhanced security protocol for Resolving stage in LISP architecture.

Chapter 5

An Enhanced Security Protocol for Resolving Addresses in LISP Network Architecture

5.1 Introduction

As stated previously, Locator/ID Separation Protocol (LISP) is a routing architecture that provides new semantics for IP addressing. To simplify routing operations and improve scalability in Internet of Things (IoT), the LISP uses two different numbering spaces spaces to separate the device identifier from its location. The Addresses Resolving in LISP architecture is a very important stage that allows the Map Server (MS) accept Map-Requests from routers, looks up database and returns the requested mapping. However, the addresses between the RLoC routers need to be addressed, e.g. when the RloC router (A) wants to send data to the RLoC router (B), both of these routers need to be authenticated so that the information can be reached from its original destination. Furthermore, LISP limits the efficiency of the security protocol which works against the redirection of the data or acting as fake routers. Therefore, this chapter provides an enhanced security protocol between the Ingress Tunnel Router (ITR) and the Egress Tunnel Router (ETR) using Challenge-Response authentication and Key agreement technique. Consequently, the structure of this chapter is divided as the following: section 5.2 discusses the first version of the designed enhanced security protocol for the Resolving stage in LISP architecture, and various discovered attacks that have been found by Automated Validation Internet Security Protocol and Applications (AVISPA) tool. Whereas, section 5.3 discusses the final version of the enhanced security protocol verified using AVISPA tool. Finally, a summary concludes the chapter in section 5.4.

5.2 First Version Enhanced Security Protocol for Resolving Procedure in LISP Architecture

This section discusses and formally analyses the security of the basic address procedure of LISP.

5.2.1 The Security Protocol Description

It has been explained in chapter 2 the transaction messages of LISP when the IoT device wants to communicate with other IoT device using the LISP ITR and ETR routers. Chapter 3 has also reviewed most of the threats weaknesses between ITR and ETR. The review has led to the realization that a security protocol is needed to secure the transfer of data between these routers. The following notations in Table 5.1 describe resolving procedure in LISP network architecture with their definitions.

Table 5.1 Notation Resolving Procedure in LISP Network Architecture

The Notation	Definition
ITR	The Ingress Tunnel Router in the source EID Space
ETR	The Egress Tunnel Router in the destination EID Spec
MS	The Map Server
N1	The Nonce
h(m)	Hash value of the message (m)
{m}{k}	The message (m) being encrypted with the Key (K).

5.2.2 The Security Protocol Exchange Messages

Figure 5.1 shows the sequence diagram for exchange message to the resolving procedure first version

$\mathbf{Step1}: \mathbf{ITR} \rightarrow \mathbf{MS}: \mathbf{ITR}, \mathbf{N1}, \mathbf{MapRequest}, \mathbf{h}(\mathbf{ITR}, \mathbf{N1}, \mathbf{MapRequest})$

The ITR sends a Map-Request message which includes a 4-byte random nonce (N1) and the address of the ITR. The ITR expects to receive the same nonce in the Map-Reply message.

$\mathbf{Step2}: \mathbf{MS} \rightarrow \mathbf{ETR}: \mathbf{ITR}, \mathbf{N1}, \mathbf{MapRequest}, \mathbf{h}(\mathbf{ITR}, \mathbf{N1}, \mathbf{MapRequest})$

The Map Server (MS) encapsulates Step 1 and passes it to the relevant ETR as Step 2.

$\mathbf{Step3}: \mathbf{ETR} \rightarrow \mathbf{ITR}: \mathbf{ETR}, \mathbf{N1}, \mathbf{MapReply}, \mathbf{h}(\mathbf{ETR}, \mathbf{N1}, \mathbf{MapRequest})$

The ETR composes Step 3 which includes a Map-Reply and the received nonce (N1). Upon receiving this message, the ITR checks the included nonce and only when the check succeeds, the ITR authenticates the ETR.

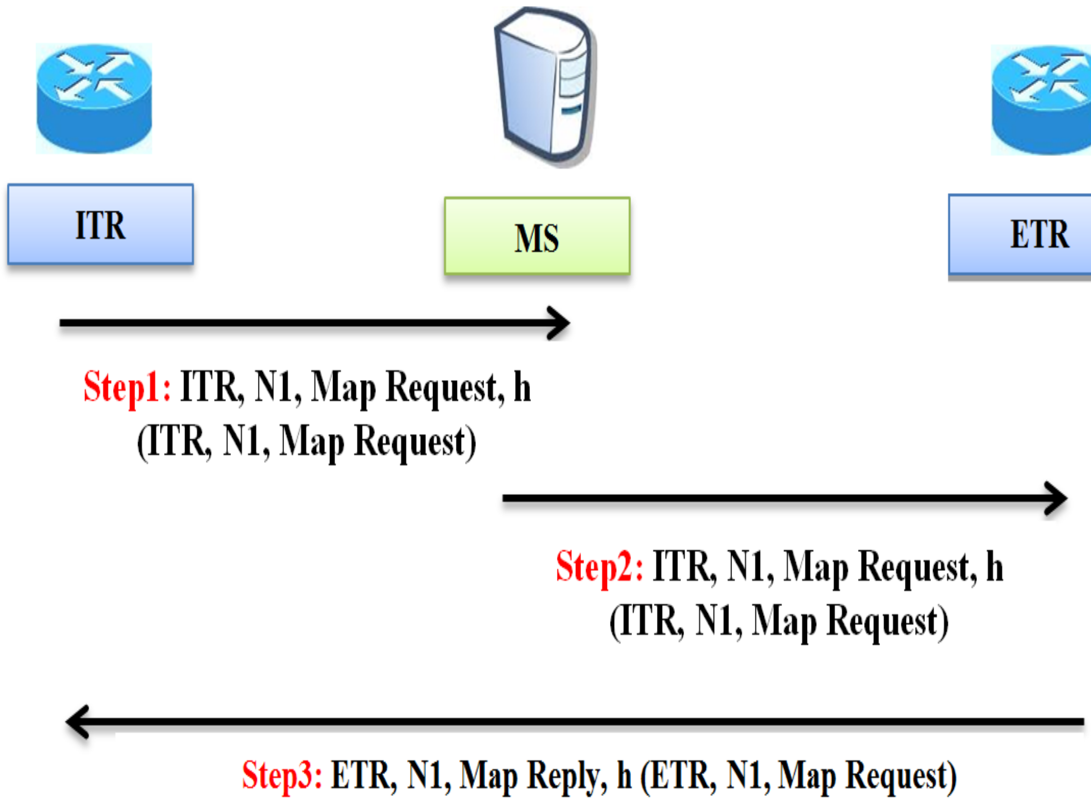


Fig 5.1 Security protocol for Resolving Procedure First Version

5.2.3 Formal Analysis of the basic Mapping Procedure using AVISPA

To analyse formally the basic mapping procedure, the system has been simulated using AVISPA tool. AVISPA file describes the system headers via using table 3.1 which has been introduced earlier in chapter 3. For conciseness, it only states here that the #specifying security Goals and the #Intruder Information heading only are described here while the rest are less significant in terms of understanding the verification process. Table 5.2 shows the security requirements of the system are defined under the # specification security Goals heading. The lines start with the keyword Secret which defines the secrecy of the protocol. The Secret _of sec_PSK_ItrMs is the PSK shared key between the ITR router and map-Server, the PSK_MsEtr is the shared key between the MS router and ETR. The N1 and N2 use the random number (nonce). Where the weak_authentication on PSK_ItrMS asserting could be interpreted as follows:

Table 5.2 HLPSL Code: Specifying Security Goals

HLPSL Code in AVISPA	**Comments**
Goal	% Specifies the security properties to be checked
secrecy_of challenge response n1 *security_of secitr, secn1,secms,secetr*	% Check the secrecy of random nonce N1 between MS and ETR.
Weak_authentication on itr_ms	% Check the authentication between ITR and MS, and then ITR should provide confirmation witness information to MS.
Weak_authentication on ms_etr	% Check the authentication between MS and ETR, and the MS should provide confirmation witness information to ETR.
Weak_authentication on itr_etr	% % Check the authentication between ITR and ETR, and the ITR should provide confirmation witness information to ETR.
authentication_on etr_itr_n1	% strong authentication between ETR and ITR on value of random nonce N1.
End goal	%End of session goal

Table 5.3 shows the # Intruder Information heading specifies the intruder identity, knowledge and capability. The first line identifies the Intruder's initial Knowledge, i.e., it has been assumed that the intruder knows the identity of the participants and can fabricate the Map Request and the Map Reply messages.

Table 5.3 Intruder Information Heading

HLPSL Code in AVISPA	**Comments**
intruder_knowledge={ITR,MS,ETR,PSK_ITRMS,PSK_MSETR,H,SK,mapRequest, mapReply}	% Specifies the intruder's knowledge and capabilities. Check the intruder between the ITR and MS on pre-shared keys PSK_ITRMS and PSKMSETR. And also check if the intruder captures the session key SK between the ETR and ITR.
composition	% It is one or more basic roles, gluing them together so they execute together, usually in parallel with interleaving semantics.
session(itr,ms,etr,PSK_ItrMs,PSK_MsEtr,SK,h)/	% set and defined ITR, MS and ETR on the network environment. Also defined the keys PSK_ItrMS, PSKEtr and SK.
session(ms,i,PSK_ItrMs,PSK_MsEtr,SK)	%Check the intruder on MS captures the PSK_ItrMS, PSK_MsEtr and Sk.
session(itr,i,sk,h)/	%Check the intruder on ITR captures the SK.
end role	%End of session role

5.2.4 Security Analysis for Resolving Procedure in LISP Network Architecture

After generating the HLPSL description of the system using AVISPA to check the security assertions, the following attacks are recognized:

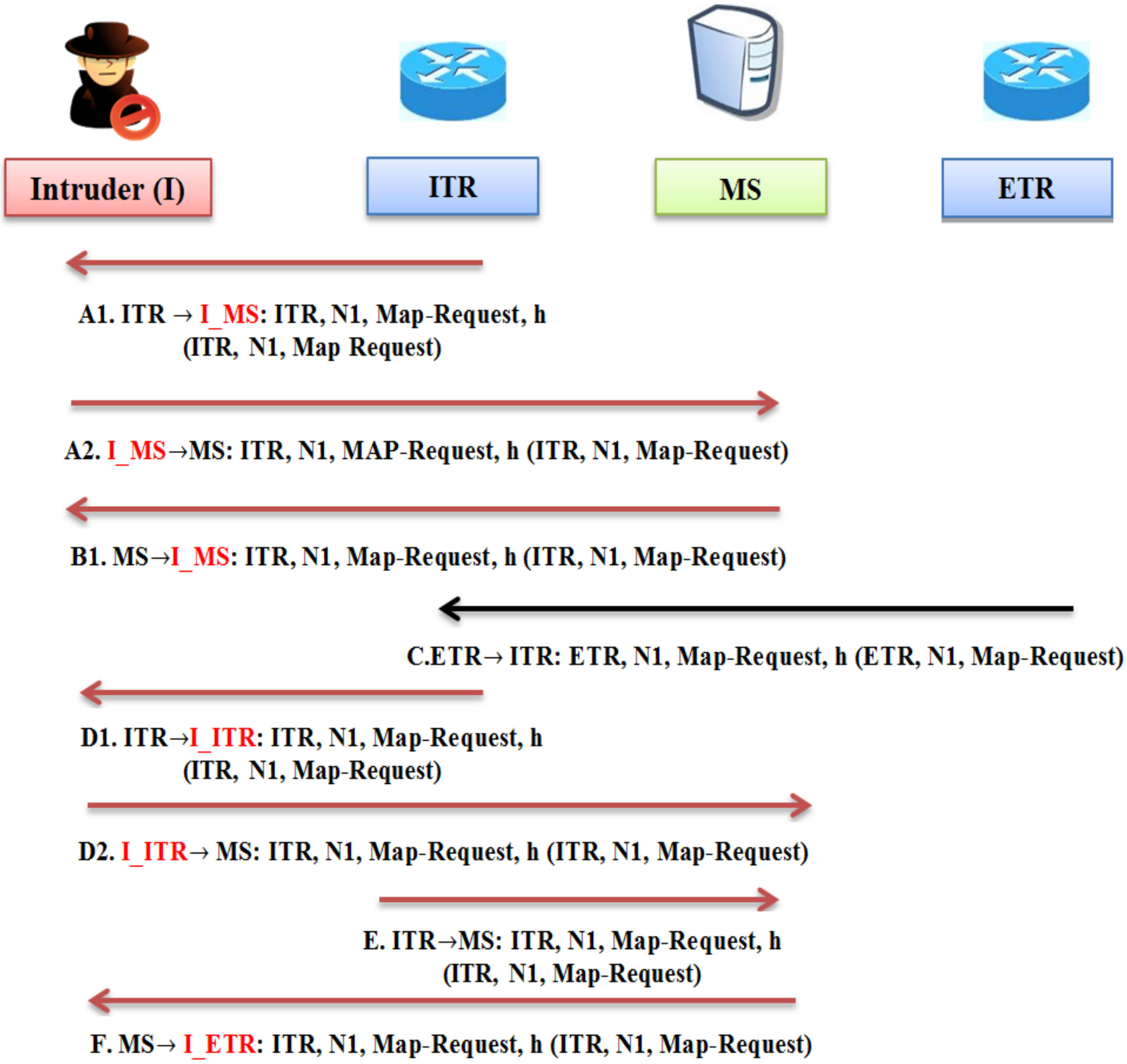


Fig 5.2 Security Attacks Discovery via AVISPA

Figure 5.2 shows attacks discovered by AVISPA; as messages (A1) to (B1) show the intruder intercepts the communication and acts as I_MS. It impersonates (imitates or mimics) both parties and gains access to information when the two parties are trying to send messages to each other. This attack is called Man in Middle Attacks (MitM), which allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late. Though, the intruder intercepts the messages (D1) to (F) and acts as I_ITR and I_ETR. This message is transformed like a hash function (h), meanwhile, the Intruder (I) eavesdrops on the conversation and keeps hash. When the interchange is over, the intruder is either as ITR or ETR. When MS asks for a proof of identity, the intruder sends ITR hash read from the first session, which MS accepts and thus grants access to the intruder. This attack is called eavesdropping attack.

5.3 Final Version Enhanced Security Protocol of the Address Resolving Procedure in LISP Network Architecture

To address the previous attacks, there is a need to secure the (ITR-MS) and the (MS-ETR) connections. As it has been mentioned in section II, for the Registration process, it is presumed that LISP-Capable router (ITR, ETR) and MS have already agreed on secret keys. Similarly, it is to be presumed that these keys will be used to secure the transactions in the resolving procedure. The protocol uses the Key Agreement technique with the challenge response. The notations in table 5.3 are used in AVISPA as the following shows:

Table 5.3 Notation of AVISPA

The Notation	Definition
ITR	The Ingress Tunnel Router in the source EID Space
ETR	The Egress Tunnel Router in the destination EID Spec
MS	The Map Server
N1,N2	The Nonce is a random number
H(m)	Hash value of the message (m)
{m}{k}	The message (m) being encrypted with the Key (K).
SK	Secret key (session Key)
PSK1	Is shared between ITR and MS
PSK2	Is shared between the MS and ETR

5.3.1 Protocol Exchange Messages

The following messages show the protocol procedures:

$$\textit{Step 1}: \boldsymbol{ITR \rightarrow MS}: \left\{ \begin{matrix} \boldsymbol{ITR, N1, MapRequest, SK,} \\ \boldsymbol{H(ITR, N1, MapRquest, SK)} \end{matrix} \right\} \{\boldsymbol{PSK1}\}$$

$$\textbf{Step 2}: MS \rightarrow ETR: \begin{Bmatrix} ITR, N1, MapRequest, SK, \\ H(ITR, N1, MapRquest, SK) \end{Bmatrix} \{PSK2\}$$

The ITR composes Step 1 and includes a freshly generated secret key (SK) to be used by the ETR to encrypt the Map-Reply packet. Then, this message is forwarded by the MS towards the ETR router in Step 2.

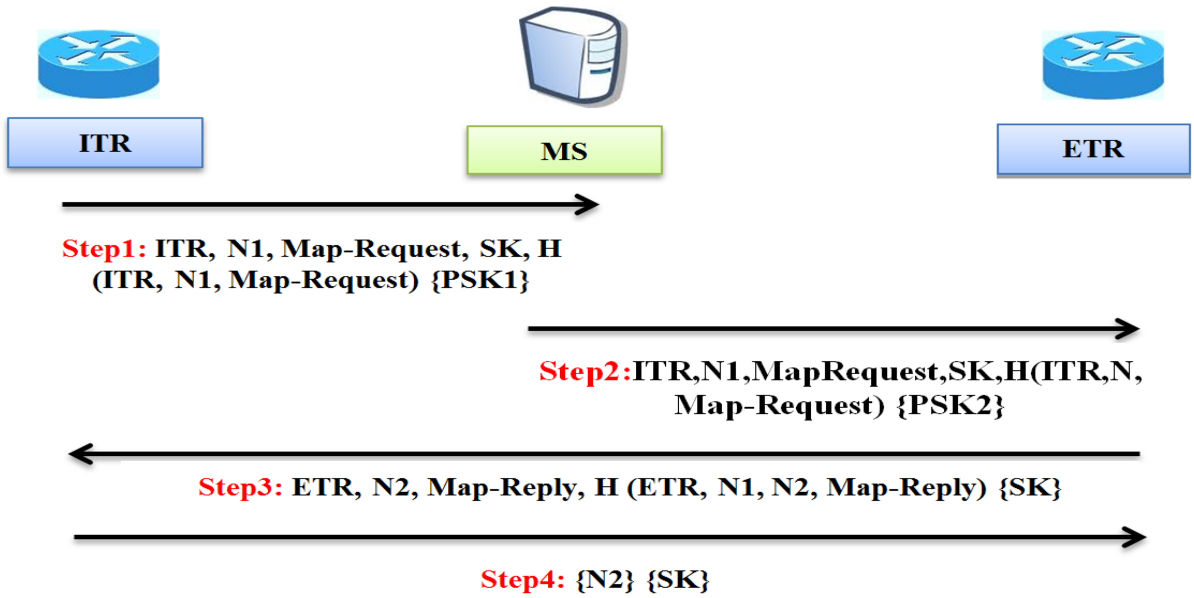


Fig 5.3 Security Protocol Address Resolving Procedure

$$\textbf{Step 3}: ETR \rightarrow ITR: \begin{Bmatrix} ETR, N2, MapReply, \\ H(ETR, N1, N2, MapReply) \end{Bmatrix} \{SK\}$$

When the ETR receives the Map-Request in Step 2, it replies a Map-Reply message in Step 3 with a challenge nonce (N2). The message is encrypted using the suggested key (SK).

$$\textbf{Step 4}: \mathbf{ITR} \rightarrow \mathbf{ETR}: \{\mathbf{N2}\}\{\mathbf{SK}\}$$

The ITR returns the challenge (N2) encrypted using the key (SK) in Step 4. The ETR then checks the returned challenge to authenticate the ITR.

5.3.2 Analysis of Results

Automated Validation of Internet Security Protocols and Applications (AVISPAP) is a modelling tool for building and analysing formal models of security protocols [AVISPA, 2013]. Figure 5.4 shows the resolving procedure security protocol resyult verified b AVISPA, which integrates four back-ends namely; the On-the-fly Model-Checker OFMC, the

Constraint-Logic-based Attack Searcher CL-AtSe, the SAT-based Model-Checker SATMC, and the TA4SP protocol analyser, which verify protocols by implementing tree automata based on automatic approximations. All the back-ends of the tool analyse protocols under the assumption of perfect cryptography and that the protocol messages are exchanged over a network that is under the control of a Dolev-Yao intruder [AVISPA, 2013]. That is, the back-ends analyse protocols by considering the standard protocol independent, asynchronous model of an active intruder who controls the network but cannot break cryptography. The intruder can inparticalar intercept messages and analyse them if he possesses the corresponding keys for decryption, and he can generate messages from his knowledge and send them under any party name.

Fig 5.4 AVISPA Results

Table 5.4 shows the results of the enhanced security protocol for resolving addresses in LISP architecture, while the input code and results are described in the appendix section -E. Besides, AVISPA is used to verify the security properties like secrecy, integrity and authentication. It gives details about whether protocol is safe or not. If not it gives the trace of the attacks found to indicate secrecy attack or authentication attack. Even through many properties of the protocol are to be checked, but only few can be verified using AVISPA. For this the modelling is done in HLPSL, language used by AVISPA tool. For our verification, OFMC, ATSE, SATMC, and TA4SP have been used to search for the attacks on the protocol. The feature of finding and tracing the attack makes AVISPA different from other tools. Hence the tool is tested and the protocol verification results are analysed; the results shows that they do not have any security flaws and the security protocol resolving addresses is safe to be used.

Table 5.4 AVISPA Tools (OFMC, ATSE, SATMC, and TA4SP) Results

Version	Tool	Description	Result
Basic session	OFMC	Visited Nodes: 287 nodes Depth: 11 plies Search Time: 0.16s	**SAFE**
Basic session	ATSE	Analysed : 18states Reachable : 6 states Translation: 0.00 seconds Computation: 0.04 seconds	**SAFE**
Basic session	SATMC	AttackFound false Boolean upper BoundReached true boolean graphLeveledOff 4 steps satSolver zchaff solver maxStepsNumber 11 steps stepsNumber 5 steps atomsNumber 545 atoms clausesNumber 1716 clause e ncodingTime 0.36 seconds solvingTime 0.006 seconds if2sateCompilationTime 0.09 seconds ATTACK TRACE %% no attacks have been found...	**SAFE & goal as specified**
Basic session	TA4SP	STALISTICS SECURITY-As specified ATTACK TRACE No attack found	**SAFE**

5.3.3 Security Analysis for the Resolving Address in LISP Network Architecture

The main goals of the proposed protocols are to achieve a mutual authentication between ETR and ITR routers and secure the direct connection between them. However, it is crucial to achieve these goals with a minimum modification to basic LISP. The security-related goals could be achieved using different protocols such as the Internet Key Exchange (IEK), Virtual Private Network (VPN) protocols and the Internet Protocol Security (IPsec) [Djeddai et al. 2016]. However, these protocols will significantly increase the number of exchanged messages, at least five extra messages in the case of IKE and more than this in the case of IPsec. Moreover, packets-encapsulation due to the tunneling process in VPN protocols will add extra headers to the LISP-capable devices. The fact that the formal verification of the enhanced protocol, using AVISPA, found no attacks against any of the checked assertions implies that the protocol achieves successfully a number of crucial security requirements such as mutual authenticating the participating parties and maintaining the security of the session key between the ITR and ETR routers. Furthermore, the protocol does not require major modification to the basic LISP transactions and no extra headers are needed for the packets encapsulation.

5.4 Summary

In this chapter, an enhanced security protocol has been introduced to secure the Resolving stage in the LISP architecture. The first designed version of the enhanced protocol uses a random value generating and functions to provide a safe and secure transaction message between ITR and ETR. The protocol is simulated using AVISPA and two attacks have been discovered namely (I) MitM and (II) eavesdropping attacks.

Therefore, the final version of the enhanced security protocol for the Resolving stage in LISP architecture has provided a new security method, based on the Challenge-Response authentication and Key agreement technique, allowing ITR and ETR to authenticate each other. The protocol is verified by AVISPA tool. The results show that this protocol is void of any security flaws.

The study goes on to propose, in chapter 6, an efficient security communication protocol for IoT based on LISP network architecture using El-Gamal encryption system. The proposed protocol method meets the objectives of practicability, simplicity and strong notions of security.

Chapter 6

A secure Authentication Protocol for Internet of Things Communication using LISP Network Architecture

6.1 Introduction

Needless to say that the Internet of Things (IoT) is becoming indeed a reality as vast numbers of smart objects are interconnected via the Internet Protocol (IP). In this context a number of applications come to handle sensitive information, and thus, security mechanisms are very much required to ensure confidentiality, integrity and authenticity of the collected information. Locator/ID Separation Protocol (LISP) is one of best solutions to adopt a huge number of the IoT devices by supporting communication and mobility of these connected devices. However, it should be always remembered that the security level of LISP is still under development. Therefore, this chapter proposes a new security protocol for IP-based Wireless Sensor (IP-WSN) using LISP architecture. The accomplished protocol is based on El-Gamal encryption system in order to provide End-to-End (E2E) Security to IoT devices. Hence, the structure of this chapter is divided as the following: 6.2 proposes the first version of the security communication protocol using Challenge-response and Diffie-Hellman key exchange and tackles different types of attacks found by Automated Validation Internet Security Protocol and Applications (AVISPA) tool. Whereas, section 6.3 presents the final version of security commotion protocol for IoT based on LISP architecture using El-Gamal encryption system verified using AVISPA tool. Finally, a summary concludes the chapter and that is in section 6.4.

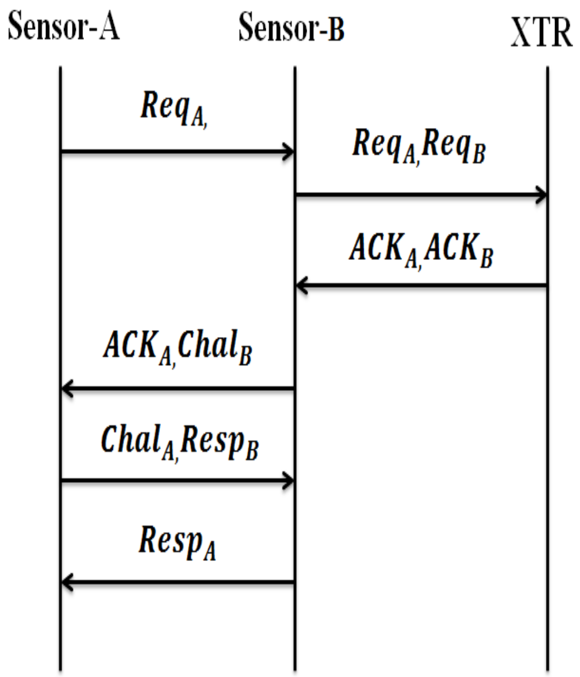
6.2 First Version of the Security Protocol for Internet of Things Communication using LISP Network Architecture

The goal of the security protocol is to offer End-to-End security to the IP-based Wireless Sensor Network (IP-WSN). Therefore, the following describes the security protocol:

6.2.1 The Security Protocol Description

In this protocol, Sensor-A wants to establish a secure connection and negotiate a Diffie-Hellman key with Sensor-B. As they do not know each other in advance, the authentication is checked and performed using an XTR router as a trusted party. Both Sensor-A and Sensor-B have initially shared keys with XTR. Figure 6.1 shows the simple exchanged messages of the protocol.

First, both Sensor-A and Sensor-B create Diffie-Hellman half-keys, along with hashes that are encrypted for XTR (denoted by the message Req_A and Req_B, respectively). After checking these messages, XTR replies with appropriate acknowledged messages ACK_A and ACK_B that also contain encrypted hashes for the respective recipients. Finally, Sensor-A and Sensor-B perform a mutual challenge-response using the new Diffie-Hellman key that is authenticated by XTR. Where, the XTR is used as a third party trust authentication in order to provide authentication to the sensor A and Sensor B, as the Diffie-Hellman performs a good keys exchange between the devices only not for authentication.



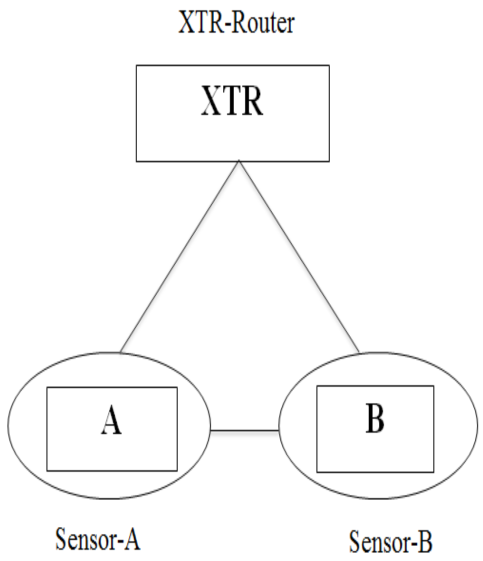


Fig 6.1 the Simple Message Exchange

Table 6.1 Notations of Communication Protocol

The Notation	Definition
Sensor-A and Sensor-B	Two communication parties, Wireless Sensor Node; its sensor device has IP address and prefixes identifying the end-points called EID.
XTR	XTR refers to a device which functions both as an Ingress Tunnel Router ITR and an Egress Tunnel Router ETR (which is usually typical),
$(\mathcal{G}, g, p)$	A finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; Both g and q are publicly known.
SK_A, SK_B	Shared Secret keys of Sensor-A and Sensor-B.
SK_{A-XTR}, SK_{B-XTR}	Shared Secret keys of Sensor-A and Sensor-B, which are shared with XTR router.
$\oplus$	Bit-wise exclusive or operation
H, H'	Two secure on-way hash functions.
DH	Diffie-Hellman
$\mathbf{CH_n}$	Challenge number n
N	Random number
$\mathbf{X_i, Y_i}$	Variable value which is chosen and computed from both XTR and Sensor nodes

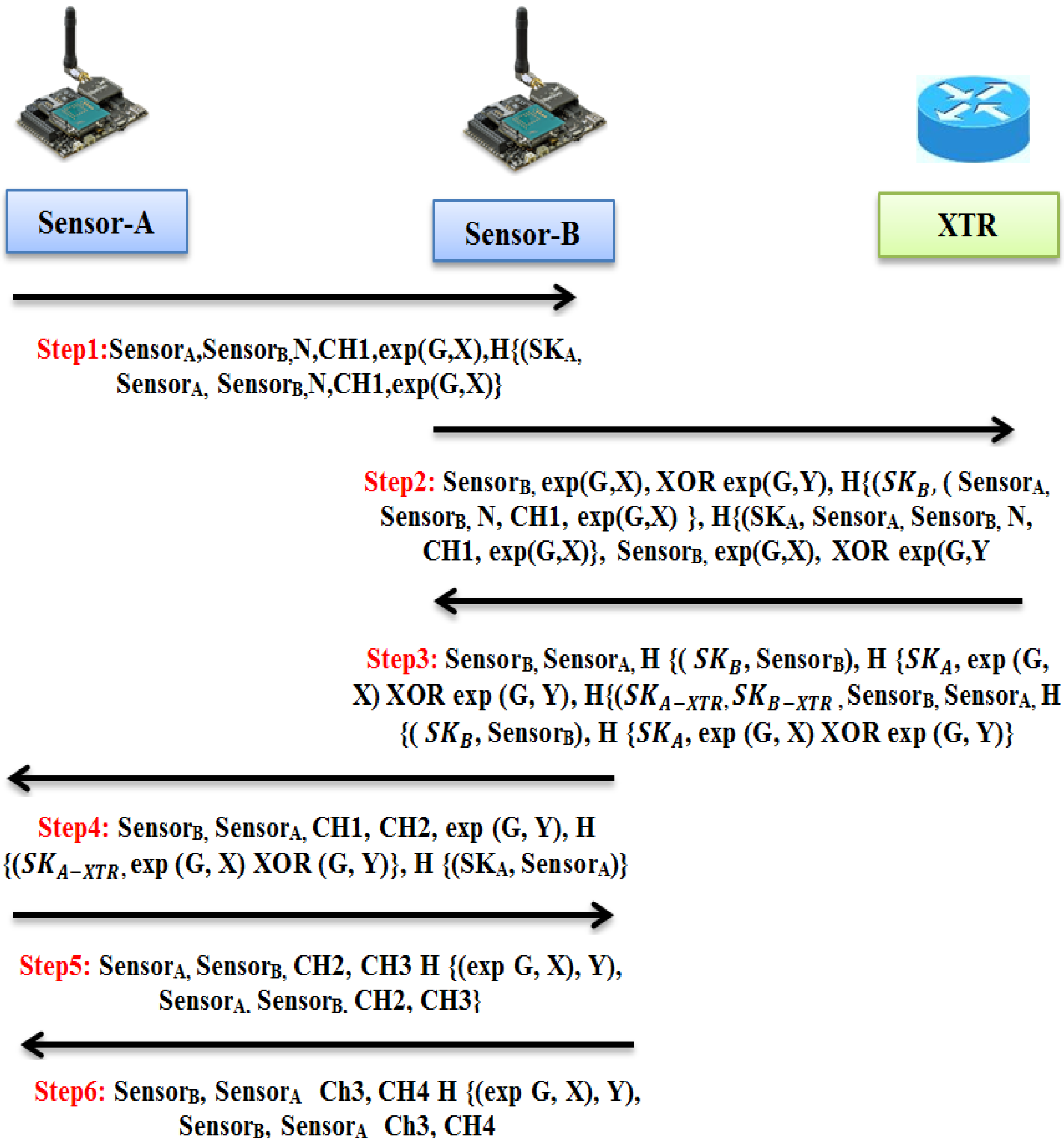
6.2.2 The Security Protocol Exchange Messages

Step 1: $\mathbf{Sensor_A \rightarrow Sensor_B : Sensor_A, Sensor_B, N, CH1, exp(G, X), H\{(SK_A, Sensor_A, Sensor_B, N, CH1, exp(G, X)\}}$

It has been assumed that two communication parties, Sensor-A and Sensor-B, want to communicate together in a secure way. In step 1, Sensor-A chooses a random X value and compute $DH: g^x mod\ p$, then it sends a challenge response CH1 and partial key of SK_A including N a nonce number and the compute function number which have computed early from Sensor-A. To prevent the adversary from revealing the message, this message is encrypted and hashed by H function.

Step 2: $\mathbf{Sensor_B \rightarrow XTR : Sensor_B, exp(G,X), XOR\ exp(G,Y), H\{(SK_B, (Sensor_A, Sensor_B, N, CH1, exp(G,X)\}, H\{(SK_A, Sensor_A, Sensor_B, N, CH1, exp(G,X)\}, Sensor_B, exp(G,X), XOR\ exp(G,Y)}$

The Sensor-B chooses a random Y value, computes $DH: g^y mod\ p$ and forwards the message to the XTR in order to verify the devices and establish authentication between the Sensor-A and the Sensor-B. This message includes the computed values of Sensor-A and Sensor-B. Obviously, both are hashed by H function. It is important to mention that the XTR is the third trust party and all the hush value and information of the devices are stored in the database of the XTR.In turn, this allows checking and verifying the device by the XTR.



F.g 6.2 The Security Communication Protocol for IoT first version

Step3: XTR → $Sensor_B$: $Sensor_B$, $Sensor_A$, H {(SK_B, $Sensor_B$), H {SK_A, exp (G, X) XOR exp (G, Y), H{(SK_{A-XTR}, SK_{B-XTR} , $Sensor_B$, $Sensor_A$, H {(SK_B, $Sensor_B$), H {SK_A, exp (G, X) XOR exp (G, Y)}

Upon receiving step2, the XTR checks verification by its database and then generates a Shared Secret keys SK_{A-XTR},SK_{B-XTR}, between the XTR and sensor nodes. These keys are usually generated after checking the computed values of the $Sensor_A$, and $Sensor_B$.

Step4: $Sensor_B$ → $Sensor_A$: $Sensor_B$, $Sensor_A$, CH1, CH2, exp (G, Y), H {(SK_{A-XTR}, exp (G, X) XOR (G, Y)}, H {(SK_A, $Sensor_A$)}

After receiving the message in Step3, the Sensor-B de-capsulate the message and computes the value that has been received from XTR and then compares it with the value that has been generated by itself in Step1 in order to verify that the original message has not been changed or manipulated by a third party. After that, if the value that has been compared is true, the Sensor-B authenticates with XTR. If the value is not true, the protocol terminates. At the same time, Sensor-B forwards a message in step 4 to Sensor-A. This message includes CH1 challenge response of Sensor-A, which has been already sent in step 1, and CH2 challenge response of Sensor-B and also SK_{A-XTR}, the shared secret key between the Sensor-A and XTR and the computed value Y of the Sensor-B encrypted with SK_A secret key of sensor-A.

Step 5: **$Sensor_A \rightarrow Sensor_B$: $Sensor_A$, $Sensor_B$, CH2, CH3 H {(exp G, X), Y), $Sensor_A$, $Sensor_B$, CH2, CH3}**

Sensor-A compares the challenge response CH1, CH2, then verifies the values which are sent from Sensor-B by computing these values and then Sensor-A will authenticate with XTR. At the same time, Sensor-A computes the secret key value that is sent from Sensor-B with the generator of XTR and then Sensor-A authenticates with sensor-B. In step 5, Sensor-A sends the computed value and its hash by H function, including also CH2 of Sensor-B and new CH3 of Sensor-A.

Step6: **$Sensor_B \rightarrow Sensor_A$: $Sensor_B$, $Sensor_A$ Ch3, CH4 H {(exp G, X), Y), $Sensor_B$, $Sensor_A$ Ch3, CH4**

Here, Sensor-B receives CH2 and CH3 in step 5. Sensor-B verifies and computes values, then Sensor-B authenticates with Sensor-A. In step 6, Sensor-B replies the message with old CH3 of Sensor-A and new CH4 of Sensor-B in order to confirm that they are now authenticated and can establish the communication safely.

6.2.3 Specification and Verification of Protocol

The first version of communication protocol for IP-WSN using LISP network architecture is simulated using AVISPA tool. In this protocol, the focus is only on the #Security Goals and the #Intruder Information heading as described here while the rest is less significant in terms of understanding the verification process. The security requirements of the system are defined under the #Specification Security Goals. The SKA and SKB are the secret shared keys which are stored in Sensor-A and Sensor-B and the shared partial between each other. The SKA-XTR, and SKB-XTR are the shared keys between the XTR and Sensor node; these keys are derived from and are generated by XTR router. The CH is the challenge response, in which one party, (i.e., Sensor-A) is a question ('Challenge') and the other party, (i.e., Sensor-B) must provide a valid answer ('response') to be authenticated. The #Intruder Information heading specifies the intruder identity, knowledge and capability. The first line identifies the intruder knowledge and defines the Intruder's initial knowledge. It has been assumed that an intruder knows the identity of the participants and fabricates the information between Sensor-A and Sensor-B or between XTR and Sensor nodes by intercepting the connection and then redirecting it to a different location.

Table 6.2 HLPSL Code: Intruder Information Heading

HLPSL Code in AVISPA	Comments
intruder_knowledge={a,b,xtr,h,g,n,ska_sk b_i_xtr}	% Specifies the intruder's knowledge and capabilities, check the intruder between Sensor-A, Sensor-B and XTR and also checks Session keys SK_A and SK_B if intruder captures between sensor-A and Sensor-B
composition	% It is one or more basic roles, gluing them together so they execute together, usually in parallel with interleaving semantics.
session(a,b,xtr,h,sk_a_xtr,sk_b_xtr,n,g)	% In the session, one usually declares all the channels used by the basic roles. Likewise, set session of Sensor-A and defined the keys $SK_{A,}$ and SK_B on the network environment.
/\session(a,b,xtr,h,sk_a_xtr,sk_b_xtr,n,g	% The /\ operator indicates that these roles should execute in parallel. In the session, one usually declares all the channels used by the basic roles. Also, it sets session of Sensor-B and defines the keys $SK_{B,}$ and SK_A on the network environment.
end role	%End of session role

It is set in this protocol as Weak_authentication on Sensor_A_Sensor_B, and Weak_authentication on Sensor_B_ Sensor_A in order to see the protocol performance and impact against the attacks. The authentication_on Router_XTR_A key, authentication_on Router_XTR_B key1 and secrecy_of sec_a_Key, sec_b_Key is set as a strong authentication on the keys generated between the XTR and Sensor nodes.

Table 6.3 HLPSL Code: Specifying Security Goals

HLPSL Code in AVISPA	Comments
Goal	% Specifies the security properties to be cheeked
weak_authentication _on sensor_a sensor_b	% Checks the authentication of Sensor-A, and then Sensor-A should provide confirmation witness information to SensorB.
weak_authentication _on sensor_b sensor_a	% Checks the authentication of Sensor-B, and the Sensor-B should provide confirmation witness information to Sensor-A.
authentication _on router_xtr_a key	% strong authentication between router XTR and Sensor-A on Key that is generated and computed by XTR router.
authentication _on router_xtr_b key1	% strong authentication between router XTR and Sensor-B on Key1 that is generated and computed by XTR router.
Secrecy_of sec_a_key, sec_b_key	Checks the secrecy between Sensor-A and Sensor-B in order to avoid any intruder between these two entities.
End goal	%End of session goal

6.2.4 Security Communication Protocol Analysis

This attack works by first observing a session between a host, (i.e., Sensor-A) and the replaying message from session Sensor-B, posed both as Sensor-A. Figure 6.3 shows the discovered attack by AVISPA tool.

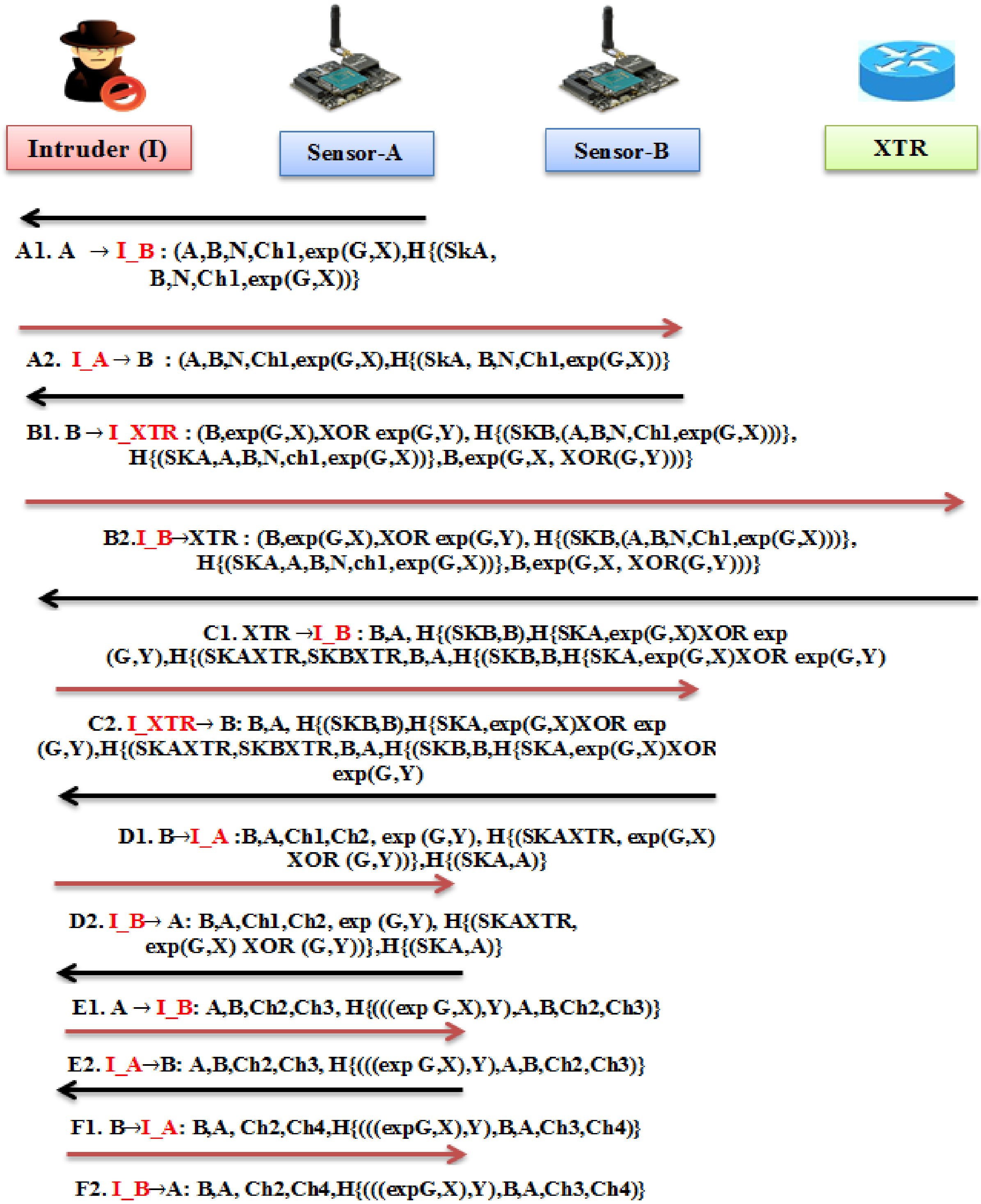


Fig 6.3 Security attacks discovery via AVISPA

Another attack recently discovered with AVISPA is the based fact that Sensor-B cannot distinguish messages (A1) to (F2) which put the security protocol on risk by spoofing the information messages exchanged between the devices. As the intruder (i) is able to play a third party, he/she can redirect the communication session to different directions. This attack is called MitM, which allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late. Therefore, a new security communication protocol is needed to provide strong authentication to sensor nodes in order to resist any possible attacks intercepting the transaction messages.

6.3 Final Version Mutual authentication Proposed for IP-WSN Communication using LISP network architecture

This section discusses an authentication and key exchange protocol using El-Gamal encryption method to secure sensor nodes communication based on the LISP protocol architecture in order to provide End-to-End Security. The protocol uses bit-wise exclusive or operation technique. In this protocol, the used notations are described as the following:

Table 6.4 Protocol Notations

The Notation	Definition
Sensor-A and Sensor-B	Two communication parties, Wireless Sensor Node; its sensor device has IP address and prefixes identifying the end-points are called EID.
XTR	XTR refers to a device which functions both as an Ingress Tunnel Router ITR and an Egress Tunnel Router ETR (which is usually typical).
$(\boldsymbol{\mathcal{G}}, \boldsymbol{g}, \boldsymbol{p})$	A finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; Both g and q are publicly known.
N	Is an element in $\mathcal{G}$
$\boldsymbol{PK_A}, \boldsymbol{PK_B}$	A public key of Sensor–A and Sensor-B stores in sensor nodes and XTR and it is shared with XTR routing.
$\boldsymbol{K_A}, \boldsymbol{K_B}$	Private Keys of Sensors-A and B, are stored in sensor nodes and are used to check and verify the computes results sent from XTR router
$\mathbf{U_A}, \mathbf{U_B}$	It is computing and generating of secret values by XTR router. This is shared between Sensor-A and Sensor-B.
$\oplus$	Bit-wise exclusive or operative.
$\boldsymbol{H'}$	Two secure on-way hash functions.
SK	Session key of (Sensor-A and Sensor-B)
$\mathbb{Z}$	Integer Number set.
$\mathbf{x_i}, \mathbf{y_i}, \mathbf{Z}, \mathbf{L}$	It is a Variable value which is chosen and computed from both XTR and Sensor nodes.
$\boldsymbol{a_i}, \boldsymbol{\beta_i}$	a_i means alpha where β_i means beta, both are used to authenticate and verify two parties together

6.3.1 Description Protocol

In this system, it has been assumed that two communication parties, Sensor-A and Sensor-B, want to communicate together in a secure way. Let PK_A be the secure key shared between Sensor-A and XTR which is arbitrary bit string. Here, Sensor-A stores (PK_A, K_A) while the XTR stores (PK_A, U_A), where U_A represents sensor-A and is derived and computed by XTR, i.e., $U_{Sensor-A} = g^{kA}$ and $(PK_A, K_A) = H(PK_A, K_A, id_{Sensor-A},)$. Similarly, for PK_B can be a secret key shared between Sensor-B and XTR. Again, Sensor-B stores (PK_B, K_B) while the XTR stores (PK_B, U_B), where $U_{Sensor-B} = g^{KB}$ and $(PK_B, K_B) = H(PK_B, K_B, id_{Sensor-B})$. Here, Sensor-A and Sensor-B can check and verify U_A and U_B which have been computed by XTR by using (PK_A, K_A) and (PK_B, K_B).

6.3.2 The Security Protocol Transaction Messages for IP-WSN Communication

Step1a: Sensor-A chooses a random number $x \in_R \mathbb{Z}_q$ and computes $(id_{Sensor-A}, PK_A)H \oplus g^x \rightarrow \mathrm{N}_{\mathrm{Sensor-A}}$, where the $id_{Sensor-A}$ is the network identity which is a portion of the TCP/IP address that is used to identify Sensor-A on the network, also the network ID here is designed to ensure the security of network and related resources. The PK_A is the public key of the sensor-A, the H is the hash function that takes the digital object passed through the algorithm to produce the hash. $\oplus$ is the XoR used for encryption and the decryption of the data. The g^x is the prime value which is publicly known, and X is the value which has been chosen from sensor-A. $\mathrm{N}_{\mathrm{Sensor-A}}$ is the outcome of the computes value. Then Sensor-A sends $(N_{Sensor-A}, id_{Sensor-A})$ to Sensor-B.

Step1b: Sensor-B chooses a random number $y \in_R \mathbb{Z}_q$ and computes $(id_{Sensor-B}, PK_B)H \oplus g^y \rightarrow \mathrm{N}_{\mathrm{Sensor-B}}$, and then Sensor-B sends $(N_{Sensor-A}, id_{Sensor-A}), (N_{Sensor-B}, id_{Sensor-B})$ to XTR router.

Step2a: Upon receiving $(N_{Sensor-A}, id_{Sensor-A})$ and $(N_{Sensor-B}, id_{Sensor-B})$, XTR uses PK_A and PK_B to compute $(id_{Sensor-B}, id_{Sensor-A}, PK_A)H \oplus \mathrm{N}_{\mathrm{Sensor-A}} \rightarrow g^x$ and $(id_{Sensor-B}, id_{Sensor-A}, PK_B)H \oplus \mathrm{N}_{\mathrm{Sensor-B}} \rightarrow g^y$ respectively. The XTR as a third trust party used its stored database to check and verify the values that have been sent from sensor A and sensor B whether they match or not.

Step2b: Now XTR chooses a random number $z \in_R \mathbb{Z}_q$ and computes $(U_{Sensor-B})^z, (U_{Sensor-A})^z, g^z \to L$, $(g^x)^z \to g^{xz} \to a$, $(g^y)^z \to g^{yz} \to b$, here the $(U_{Sensor-B})^z and\ (U_{Sensor-A})^z$ are the XTR shared key between Sensor-A and Sensor-B, Z is the chosen value of XTR router. The g is the publicly known prime value. L is the outcome of the computes value. Where a represents sensor-A which has been computed in step1a, and b represents sensor-B which has been computed in step1b. Then XTR computes $((U_{Sensor-A})^z, g^x, id_{Sensor-B,}id_{Sensor-A,}PK_A)H \oplus b \to Z_{Sensor-A}$ and $((U_{Sensor-B})^z, g^y, id_{Sensor-B,}id_{Sensor-A,}PK_B)H \oplus a \to Z_{Sensor-B}$ and XTR sends $(Z_{Sensor-A}, L), (Z_{Sensor-B}, L)$ to Sensor-B.

Step3a: Once Sensor-B receives the sent message, it uses K_B the private key to compute $L^{KB} \to (U_{Sensor-B})^z$ which is the variable value chosen and computed from the XTR router, and it is used as shard key between the XTR and Sensor-B. As the same time it used K_B to compute $((U_{Sensor-B})^z, g^y, id_{Sensor-B}, id_{Sensor-A}, PK_B)H \oplus Z_{Sensor-B} \to a$ and sensor-B authenticates with XTR. Now, Sensor-B uses y which is the previously chosen value to compute $a^y \to g^{xyz} \to K, (K, id_{Sensor-B}, id_{Sensor-A})H \to \alpha$ and forwards $(Z_{Sensor-A}, L), \alpha$ to Sensor-A.

Step3b: Sensor-A receives $(Z_{Sensor-A}, L, \alpha)$ and it uses K_A to compute $L^{KA} \to (U_{Sensor-A})^z$ and $((U_{Sensor-A})^z, g^x, id_{Sensor-B}, id_{Sensor-A}, PK_A)H \oplus Z_{Sensor-A} \to b$ and authenticates the XTR router. Then, Sensor-A uses x to compute $b^x \to g^{xyz} \to K$ and checks whether $(K, id_{Sensor-B}, id_{Sensor-A})H \to \alpha$ holds or not. If it does not hold, the protocol terminates, Otherwise, Sensor-A is convinced that K is a valid session key. After that, Sensor-A computes $(K, id_{Sensor-B}, id_{Sensor-A})H \to \beta$ and forwards it to Sensor-B. Sensor-A computes the Session Key $(K, id_{Sensor-B}, id_{Sensor-A})H' \to Sk_{Sensor-A}$.

Step3c: Upon receiving β , Sensor-B computes $(K, id_{Sensor-B}, id_{Sensor-A})H \to \beta$ and verifies whether computed β is equal to the received β. If both are equal, then B authenticates Sensor-A and computes the session key $(K, id_{Sensor-B}, id_{Sensor-A}\ ) \to Sk_{Sensor-B}$

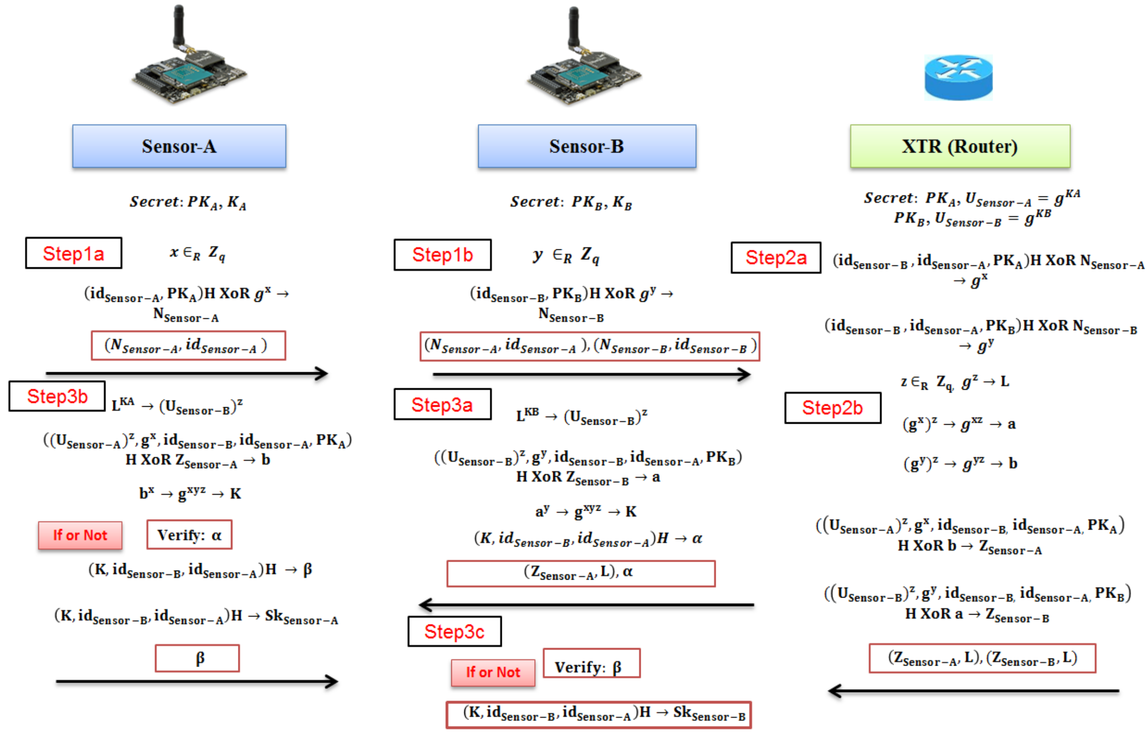


Fig 6.4 the proposed Security Protocol for IP-WSN node Communication using LISP Network

6.3.3 Security Protocol Analysis for IP-WSN Communications

The main goal of the proposed protocol is to provide mutual authentication and E2E security between the IP-WSN nodes, (i.e., Sensor-A and Sensor-B) and XTR router when the devices are communicating with each other. Therefore, this section provides a security protocol proposed analysis for IP-WSN communications.

6.3.3.1 Trivial Attacks:

Actually, computing the session key from the transmitted messages α and β is impossible due to one-way of hash function and, also, for computing it from other transmitted messages. The latter can $Z_{Sensor-A}$ or $Z_{Sensor-B}$ where an attacker has to face the difficulty of a discrete logarithm problem. Consequently, this protocol is resistant to trivial attack.

6.3.3.2 Secret Keys Guessing Attacks:

Suppose an attacker or malicious node Sensor-B tries to guess Sensor-A secret key as PK_A generates $(id_{Sensor-B,}id_{Sensor-A,}PK_A)H \oplus N_{Sensor-A} \rightarrow g^x$ and the XTR router in online Message 1 of the protocol. To verify the correctness of his guessed secret key; it needs to compute $((U_{Sensor-A})^z, g^x, id_{Sensor-B,}id_{Sensor-A}, PK_A)H \oplus Z_{Sensor-A} \rightarrow b$

and $((U_{Sensor-b})^z, g^y, id_{Sensor-B}, id_{Sensor-A,}PK_A)H \oplus Z_{Senosr-B} \rightarrow a$ as it needs the values of K_A and K_B for computing $(U_{Sensor-A})^z$ and $(U_{Sensor-B})^z$. Similarly, remaining off-line also and using the transferred messages $N_{Sensor-A}$ $N_{Sensor-B}$, $Z_{Sensor-A}$, L, an attacker cannot verify the correctness of its guessed secret key.

6.3.3.3 Man in the Middle Attack:

In message 2 of the protocol, XTR authenticates the two communicating parties Sensor-A and Sensor-B from the message $(id_{Sensor-A}, PK_A)H \oplus g^x \rightarrow \mathrm{N_{Sensor-A}}$, and $(id_{Sensor-B}, PK_B)H \oplus g^y \rightarrow \mathrm{N_{Sensor-B}}$, sent by Sensor-B. Sensor-A and Sensor-B authenticate XTR, from $((U_{Sensor-A})^z, g^x, id_{Sensor-B,}id_{Sensor-A,}PK_A)H \oplus b \rightarrow Z_{Sensor-A}$ and $((U_{Sensor-B})^z, g^y, id_{Sensor-B,}id_{Sensor-A,}PK_B)H \oplus a \rightarrow Z_{Sensor-B}$ as PK_A , PK_B are known only to XTR. Finally, Sensor-A authenticates Sensor-B from $(K, id_{Sensor-B}, id_{Sensor-A})H \rightarrow \alpha$. Based on this, each party authenticates the other communicating party and, hence, there is no scope for MitM.

6.3.3.4 Replay Attack:

Since one way-hashed function is used, this protocol is invulnerable to this attack.

6.3.3.5 Perfect Forward Security:

When the secret keys of PK_A and PK_B of Sensor-A and Sensor-B devices are compromised, the attacker cannot calculate the session key as K_A and K_B as known. These values remain unknown even to the XTR router, so there is no chance of any compromise. Also, the session key is independent on any session and x, y, z are randomly chosen.

However, the security-related goals could be archived using different protocols. For example, there are the Internet key Exchange (IKE), the virtual Private Network (VPN) protocols and the Internet Protocol Security (IPSec) as mentioned above. Furthermore, packets encapsulation due to the tunnelling process in VPN protocols will add extra load to the header of Sensor communication packets which make them incompatible with the current implementation Sensor communication capable devices.

6.3.4 Specification and Verification of Protocol

As mentioned earlier, the proposed protocol has been implemented and evaluated using AVISPA tool. The achieved result has shown that no attack is being found. For this protocol, three basic roles played by Sensor (A), Sensor (B) and XTR (R) router have been defined. K_A and K_B are secret keys of Sensor-A and Sensor-B. PK_A and PK_B are shared with XTR and hence represent the symmetric keys . K_A and K_B remain secret with Sensor-A and Sensor-B as their private keys. XTR router gets $U_A = \exp(G, K_A)$ from Sensor-A and $U_B = \exp(G, K_B)$ from Sensor-B. Hence U_A and U_B are the public keys whose inverse is known only to Sensor-A and Sensor-B respectively.

Table 6.5 HLPSL Code: Basic Roles

HLPSL Code in AVISPA	**Comments**
role sensor_A(A, B, XTR : agent, PKA : symmetric_key, SND, RCV : channel(dy), H : hash func, G : text) played_by A def= local State : nat, X, Z : text, UA : public_key, GY , Key, L : message, const sec_m_Key : protocol_id, init State := 0	% This is (a fragment of) a role as Sensor_A, with parameters A, B and XTR type agent, and Symmetric_key. The RCV and SND parameters are type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel. The parameter A appears in the played_by section, which means, intuitively, that A denotes the name of the agent who plays role as Sensor_A. Furthermore, the local section which declares local variables of Sensor_A: in this case, one called State which is a nat (a natural number) and another called UA, which will represent the public key. The local State variable is initialised to 0 the init section.
role sensor_B (A, B, XTR : agent, PKA, PKB : symmetric_key, SND, RCV : channel(dy), H : hash func, PKA, PKB : symmetric key, G :text) played by B def= local State : nat, X, Y, Z : text, GX, GY : message, UB : public key, Key : message, M1:hash(symmetric key.agent.agent.message.message).message, M2: hash(symmetric-key.agent.agent.message.message).message const sec v Key : protocol id init State := 0	% The role is as Sensor_B, with parameters A, B and XTR type agent, and Symmetric_key. The RCV and SND parameters are type channel, indicating that the channel type, in this cause (dy), denotes the intruder model to be considered for this channel. The parameter B appears in the played_by section, which means, intuitively, that B denotes the name of the agent who plays role Sensor_B. Besides, note the local section which declares local variables of Sensor_B: in this case, one called State which is a nat (a natural number) and another called UB, which will represent the public key. The M1 is represented on H(PKA.A.B.exp(G, X).exp(UA, Z)).exp(L, Z), where M2 is represented on H(PKB.A.B.exp(G, Y).exp(UB, Z).exp(L, Z). The local State variable is initialised to 0 in the init section
role router_xtr(A, B, XTR : agent, PKA, PKB : symmetric_key, SND, RCV : channel(dy), H : hash func, PKA, PKB : symmetric key, G : text) played by XTR def= local State : nat, X, Y, Z : text, UA, UB : public key GX, GY : message init State := 0	% The router_xtr represent the role with parameters A,B, and, and Symmetric_key. The RCV and SND parameters are type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel. The parameter XTR appears in the played_by section, which means, intuitively, that XTR denotes the name of the agent who plays role router_xtr. Besides, note the local section which declares local variables of router_xtr: in this case, one called State which is a nat (a natural number) and another called UA, UB which will represent the public keys. The local State variable is initialised to 0 the init section

After defining the #basic roles, it is essentially needed to define the composed roles which describe the sessions of the protocol. The #composed roles have no transition section, but rather a composition section in which the basic roles are instantiated. The /\ operator indicates that these roles should execute in parallel. In the *#session role*, it usually declares all the channels used by the basic roles. These variables are not instantiated with concrete constants. The *channel* type takes an additional attribute, in parentheses, which specifies the intruder model that is assumed for that channel. Here, the type of the declaration *channel (dy)* stands for the Dolev-Yao intruder model [AVISPA, 2013]. Under this model, the intruder has full control over the network, i.e., all messages sent by agents will go to the intruder. The latter may intercept, analyse and/or modify messages (as far as he knows the required keys) and may send any message he composes to whoever he pleases, posing as any other agents. Finally, a top-level role is always defined. This role contains global constants and a composition of one or more sessions, where the #intruder may play some roles as a legitimate user. There is also a statement which describes what knowledge the intruder initially has. Typically, this includes the names of all agents, specifically all the symmetric keys and any shares with others. It is to be noticed that, the constant 'I' is used to refer to the intruder as the source code below:

Table 6.6 HLPSL Code: Role Session

HLPSL Code in AVISPA	Comments
role session(A, B, XTR : agent, *H : hash func,* *PKA, PKB : symmetric key,* *UA, UB :public key,* *G : text)* *def=* *local SND, RCV : channel (dy)* *composition* *sensor_a(A,B,XTR,SND,RCV,H,PKA,G)* */\ sensor_b(A,B,XTR,SND,RCV,H,PKA,PKB,G)* */\ router_xtr(A,B,XTR,SND,RCV,H,PKA,PKB,G)* *end role*	% role session is a basic role, gluing them together so they execute together, usually in parallel with interleaving semantics. The /\ operator indicates that these roles should execute in parallel. In the session role, one usually declares all the channels used by the basic roles. These variables are not instantiated with concrete constants. The channel type takes an additional attribute, in parentheses, which specifies the intruder model one assumes for that channel. Here, the type declaration channel (dy) stands for the Dolev-Yao intruder model. Under this model, the intruder has full control over the network, that all messages sent by agents will go to the intruder. He may intercept, analyse, and/or modify messages (as far as he knows the required keys), and send any message he composes to whoever he pleases, posing as any other agent.
intruder knowledge = {a, b, xtr, g, h, pi, ua, ub, ui} *composition* *session(b, a, xtr, h, pa, pb, ua, ub, g)* */\ session(i, b, xtr, h, pi, pb, ui, ub, g)* */\ session(a, i, xtr, h, pa, pi, ua, ui, g)* *end role*	% Specifies the intruder's knowledge and capabilities. Check the intruder between the Sensor-A, Sensor-B and XTR. And also check Symmetric_key (pa,pb,pi)and Public(ua,ub,ui) key if intruder captures between sensor-A and Sensor-B and XTR. The /\ operator indicates that these roles should execute in parallel. End of session role

#Specifying Security Goals are specified in HLPSL by enhancing the transitions of the basic roles with the so-called goal facts and by then assigning them a meaning by describing, in the HLPSL *goal* section, what conditions, i.e., what combination of such facts indicate an attack and a violation of *secrecy*. The goal declaration section describes that it should be considered as an attack when the intruder learns a secret value internally, the attack conditions are specified in terms of a temporal logic. Also, useful and concise macros are provided for two of the most frequently used security goals, authentication and secrecy.

Table 6.7 HLPSL Code: Specifying Security Goals:

HLPSL Code in AVISPA	**Comments**
Goal	% Specifies the security properties to be cheeked
authentication on key	% strong authentication between router XTR and Sensor-A on Key that is generated and computed by XTR router
authentication on key1	% % strong authentication between router XTR and Sensor-B on Key1 that is generated and computed by XTR router.
secrecy-of sec-m-Key, sec-v-Key	Check the authentication of generated and computed keys by XTR
End goal	%End of session goal

6.3.5 Analysis of Results

In this section the communication protocol is tested by Automated Validation of Internet Security Protocols and Applications (AVISPAP). It has used four back-end tools which are integrated by AVISPA, namely; the On-the-fly Model-Checker OFMC, the Constraint-Logic-based Attack Searcher CL-AtSe, the SAT-based Model-Checker SATMC and the TA4SP protocol analyser, which verifies protocols by implementing tree automata based on automatic approximations. All the back-ends of the tool analyse protocols under the assumptions of perfect cryptography and that the protocol messages are exchanged over a network that is under the control of a Dolev-Yao intruder [AVISPA, 2013]. That is, the back-ends analyse protocols by considering the standard protocol independent, asynchronous model of an active intruder who controls the network but cannot break cryptography; in particular, the intruder can intercept messages and analyse them if he possesses the corresponding keys for decryption, and he can generate messages from his knowledge and send them under any party name as figure 6.5 shows.

Fig 6.5 AVISPA Results

Table 6.8 shows the results of the security communication protocol for IoT based on LISP architecture, while the input code and results are described in the appendix section -F. Significanly, AVISPA is used to verify the security properties like secrecy, integrity and authentication. It gives details about whether protocol is safe or not. If not it gives then the trace of the attacks found, to indicate secrecy attack or authentication attack. So even though many properties of the protocol are to be checked, but only few can be verified using AVISPA. For this the modelling is done in HLPSL, language used by AVISPA tool. For our verification, OFMC, ATSE, SATMC, and TA4SP have been used to search for the attacks on the protocol. The feature of finding and tracing the attack makes AVISPA different from other tools. Henceforth, the tool is tested and those protocol verification results are analysed. The results show that there are not any security flaws and the security communication protocol is safe to be used.

Table 6.8 AVISPA Tools (OFMC, ATSE, SATMC, and TA4SP) Results

Version	Tool	Description	Result
Basic session	OFMC	VisitedNodes:25674 nodes Depth: 6 plies Search Time: 0.2s	**SAFE**
Basic session	ATSE	Analysed: 3874 States Reachable: 2635 States Translation: 0.00 seconds Computation: 0.01 seconds	**SAFE & goal as specified**
Basic session	SATMC	STATISTICS Attack Found : false Boolean Upper Bound Reached: true Boolean Graph Leveled off: 5 steps Sat Solver: zchaff Solver Max Steps Number: 11 Steps Steps Number : 5 Steps Atoms Number: 543 Atoms Clauses Number 1613 Clauses Encoding Time: 0.2 Seconds If2Sate Compilation Time 0.02 Seconds ATTACK TRACE %%no attacks have been found…	**SAFE**
Basic session	TA4SP	STALISTICS SECURITY-As specified ATTACK TRACE No attack found	**SAFE**

6.4 Summary

As demonstrated in this chapter, a security communication protocol for IoT based on LISP was proposed in two versions. The first version of the security protocol used Challenge-response and Diffie-Hellman key exchange; it was proposed to secure the communication between the IoT devices, when they communicate with each other based on LISP network architecture. A formal analysis was provided using AVISPA tool and Man in the Middle Attack (MitM) was discovered via AVISPA security tool. Furthermore, the final version of a new security protocol for IP-based Wireless Sensor (IP-WSN) communication has been proposed using El-Gamal encryption system. The security analysis and verification using AVISPA show that no attacks were found in the communication protocol.

In chapter 7, an interface is set on each level of the protocol in order to achieve secure refinement protocol to the IoT-based on LISP network architecture. Certainly, these proposed protocols methods meet the targeted objectives of practicability, simplicity and the strong notions of security.

Chapter 7:

A Security Refinement Protocol and Performance Analysis for Internet of Things Communication using LISP Network Architecture

7.1 Introduction

To begin with, chapter 4 has revealed various numbers of vulnerabilities in Internet of Things (IoT) devices: revealing these vulnerabilities enables the adversary to capture a node easily. Consequently, this chapter proposes security refinement to IoT devices using Locator/ID Separation Protocol (LISP). Besides, an interface is set as encapsulated security protocols namely resignation, resolving and communication protocols to accomplish a robust refinement to IoT at network layer. Furthermore, the security refinement protocol is verified using Automated Validation Internet Security Protocol and Applications (AVISPA) and the achieved results show that they do not have any security flaws. The chapter also provides performance security refinement protocol analysis and evaluation for IP-Based Sensor Networks (IP-WSN) communication using Contiki and Cooja simulation tool. Therefore, the chapter is organized as follows: Section 7.2 proposes a security refinement protocol interface. Section 7.3 discusses the simulation and performance analysis. Section 7.4 discusses the analysis Performance of communication overhead. Section 7.5 discusses the performance analysis of power computation for the security refinement protocol. Section 7.6 discusses the performance analysis of memory consumption. Section 7.7 discusses the performance analysis of energy consumption. Section 7.8 discusses the security refinement protocol resilience against node compromise. Finally, section 7.9 concludes the chapter and summarises the main points.

7.2 Security Refinement Protocol for Internet of Things using LISP Network Architecture

This section introduces security refinement protocol architecture to sensor node based on LISP network architecture in order to provide authentication and key agreement at network levels.

The next figure shows the keys tree that is used for the security purpose in this refinement protocol. Here, the private key (K_i) and public key (PK_i) are stored in the Sensor device, which is used to decrypt and verify the authentication process. The keys PK1, PK2, PK3 and PK4 are used for encrypting the communication channel whereas SK is the session key used as key agreement between the parties at the network levels, while section 7.2.1 will explain the messages exchange of refinement protocol for IoT devices using LISP architecture.

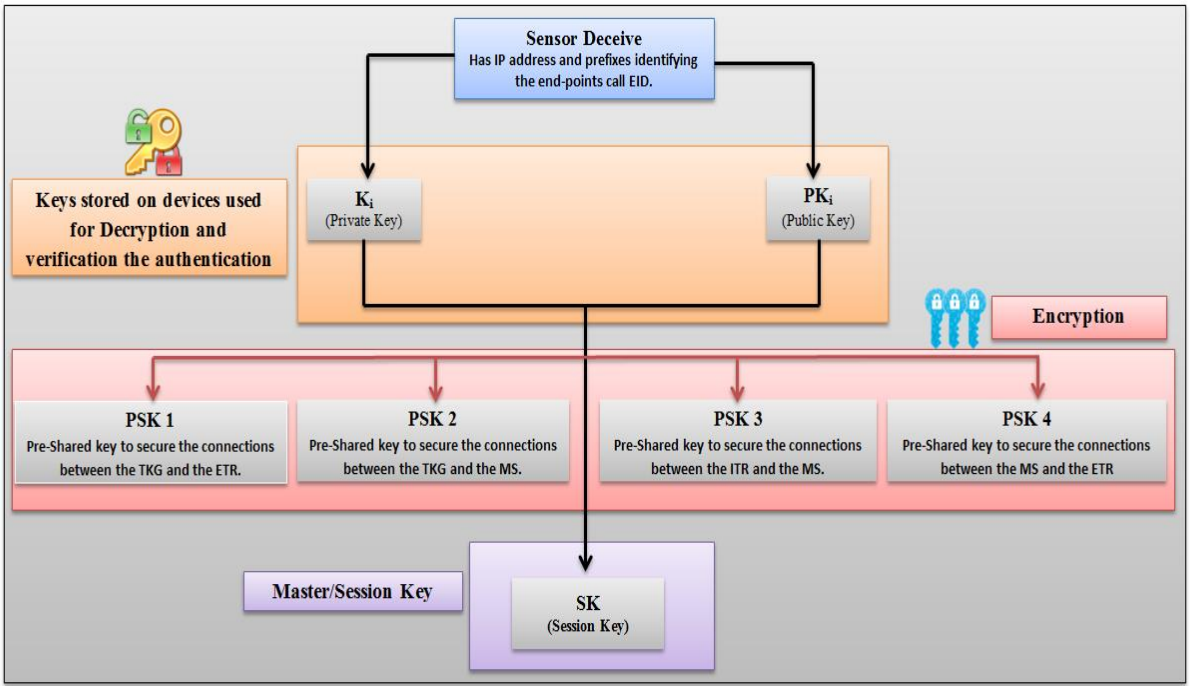


Fig 7.1 Keys Tree for Security refinement Protocol

Table 7.1 Protocol Notation of Refinement Protocol

The Notation	Description
TKG	The Trusted Ticket Granting; it randomly generates pairs of Public/Private keys.
$Sensor_i$	Wireless Sensor Node; it is a sensor device that has IP address and prefixes identifying the end-points called EID.
ITR	The Ingress Tunnel Router in the source EID Space
ETR	The Egress Tunnel Router in the destination EID Space
XTR	Refers to the device which functions both as Ingress Tunnel Router ITR and an Egress Tunnel Router ETR.
MS	The Map Server.
n_i	The Nonce is a random number
K(ETR), K(MS)	The private keys of the ETR and MS respectively. These keys are derived by the TKG.
PSK1	Pre-Shared key to secure the connection between the TKG and the ETR.
PSK2	Pre-Shared key to secure the connection between the TKG and MS.
PSK3	Pre-Shared Key between ITR and MS.
PSK4	Pre-Shared Key between the MS and ETR
PK_i	Public key
K_i	Private key

U_i	It is a partial of private key shared with XTR and it is derived from and computed by XTR Router.
$(\mathcal{G}, \mathbf{g}, \mathbf{p})$	A finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; both g and q are publicly known.
$\oplus$	Bit-wise exclusive or operation
SK	Session Key
N	Is an element in $\mathcal{G}$
$\mathbb{Z}$	Integer Number set
x,y,Z,L	It is a variable value which is chosen and computed from both XTR and Sensor Node
$\boldsymbol{\alpha_i}, \boldsymbol{\beta_i}$	α_i means alpha where β_i means beta; both are used to authenticate and verify two parties together.
H(m)	Hash value of the message (m)
{m}{k}	The message (m) being encrypted with the Key (K).

7.2.1 The Security Refinement Protocols Transaction Messages for Internet of Things

Phase 1

Step0: $\mathbf{ETR \rightarrow EID: Adv}$

The protocol starts when ETR detects the new host (EID) on the LISP network architecture domain.

Step1: $\mathbf{TKG \rightarrow ETR : \{\, K(ETR)\{PSK1\}}$

Step1: $\mathbf{TKG \rightarrow MS : \{\, K(MS)\{PSK2\}}$

In the Step1 and Step2, the Trusted Ticket Granting (TKG) responds by providing two communicating parties (ETR, MS) with their keys $\mathrm{K(ETR), K(MS)}$. These two messages are encrypted using the pre-shared secret keys $\mathrm{PSK1}$, and $\mathrm{PSK2}$, respectively.

Step3: $\mathbf{ETR \rightarrow MS: \{Map - Register\}\{PK(MS)\}, \{h(Map - Register)\}\{K(ETR)\}}$

Here, the ETR sends LISP Map-Register packet in Step3. The content of this packet is encrypted using the MS's Public key, which is publicly known and digitally signed using the private key of the ETR, the Map-Register packet includes the ETR's address (RLOC), a random (n1) and a list of EID-Prefix managed by ETR.

Step4: $\mathbf{Ms \rightarrow ETR: \{Map - Notify\}\{PK(ETR)\}, \{h(Map - Notify)\}\{K(MS)\}}$

Upon receiving Step3, the MS will use its private key K(MS) to decrypt the message and then verify the signature using the ETR's public key $\mathrm{PK(ETR)}$. Finally, the MS will hash the included Map-Register and compare the result with the received signed value. Only if the two values are equal, the MS composes a Map-Notify packet as Step4, which includes the received random number (n1). This message is encrypted using the ETR's public key and digitally random number (n1). The same message is encrypted using the ETR's public key and digitally signed using the MS's private key. The ETR will check the included random number and only when the check succeeds, the ETR authenticates the MS.

Phase 2

Step5: $\mathbf{ITR \rightarrow MS: \{ITR, N1, MapRequest, SK, H(ITR, N1, MapRquest, SK,)\}\{PSK3\}}$

The registration protocol will send a confirmation message that registration has been done successfully and ETR has been authenticated with MS in Step4. The second protocol will be run automatically in order to process authentication between the routers, i.e., ITR and ETR. Otherwise, the protocol will terminate. In Step6, the ITR will send a map-Request message which includes 4 byte random nonce (N1) and the address of the ITR. The ITR expects to receive the same nonce in the Reply message. In addition, Step5 includes a freshly generated Secret (SK) to use the ETR and then to encrypt the Map-Reply packet.

Step6: $\mathbf{MS \rightarrow ETR: \{ITR, N1, MapRequest, SK, H(ITR, N1, MapRquest, SK)\}\{PSK4\}}$

The MS forwards the message in Step6 towards the ETR router.

Step7: $\mathbf{ETR \rightarrow ITR: \{ ETR, N2, MapReply, H(ETR, N1, N2, MapReply)\}\{SK\}}$

Upon receiving Step6, the ETR replies a Map-Reply message in Step7 with a challenge nonce (N2). The message is encrypted using the suggested key (SK).

Step8: $\mathbf{ITR \rightarrow ETR: \{N2\}\{SK\}}$

The ITR returns the challenge (N2) encrypted using the key (SK) in Step8 the ETR, then, checks the returned challenge to authenticate the ITR.

Phase 3

Step9A: When Sensor-A device wants to establish communication with anther device Sensor-B on different LISP network architecture domain, the third communication protocol will run automatically if the second protocol of resolving addresses has been successfully preceded and ITR has been authenticates with ETR in Step8. This will be done by sending a confirmation message from the second protocol that routers authenticate successfully, otherwise the protocol will terminate. Here, Sensor-A chooses a random number $x \in_R \mathbb{Z}_q$ and computes $(id_{Sensor-A}, PK_A)H \oplus g^{x} \rightarrow \mathrm{N}_{\mathrm{Sensor-A}}$, where the $id_{Sensor-A}$ is the network identity which is a portion of the TCP/IP address that is used to identify Sensor-A on the network. The network ID here is designed to ensure the security of the network and related resources. The PK_A is the public key of sensor-A and the H is the hash function that takes the digital object passed through the algorithm to produce the hash.

$\oplus$ is the XoR used for encryption and the decryption of the data. The g^{x} is the prime value which is publicly known, and X is the value which has been chosen from sensor-A. $N_{\text{Sensor}-A}$ is the outcome of the computes value. Then Sensor-A sends $(N_{Sensor-A}, id_{Sensor-A})$ to Sensor-B.

Step9B: Sensor-B chooses a random number $y \in_R \mathbb{Z}_q$ and computes $(id_{Sensor-B}, PK_B)H \oplus g^{y} \rightarrow N_{\text{Sensor}-B}$, and then Sensor-B sends $(N_{Sensor-A}, id_{Sensor-A}), (N_{Sensor-B}, id_{Sensor-B})$ to XTR router.

Step10A: Upon receiving $(N_{Sensor-A}, id_{Sensor-A})$ and $(N_{Sensor-B}, id_{Sensor-B})$, XTR uses PK_A and PK_B to compute $(id_{Sensor-B}, id_{Sensor-A}, PK_A)H \oplus N_{\text{Sensor}-A} \rightarrow g^{x}$ and $(id_{Sensor-B}, id_{Sensor-A}, PK_B)H \oplus N_{\text{Sensor}-B} \rightarrow g^{y}$ respectively. The XTR as a third trust party used its stored database to check and verify the values that have been sent from sensor A and sensor B whether they match or not.

Step10B: Now XTR chooses a random number $z \in_R \mathbb{Z}_q$ and computes $(U_{Sensor-B})^z, (U_{Sensor-A})^z, g^z \rightarrow L$, $(g^x)^z \rightarrow g^{xz} \rightarrow a$, $(g^y)^z \rightarrow g^{yz} \rightarrow b$, here the $(U_{Sensor-B})^z and\ (U_{Sensor-A})^z$ are the XTR shared key between Sensor-A and Sensor-B, Z is the chosen value of XTR router. The g is the publicly known prime value. L is the outcome of the computes value. Where a represents sensor-A which has been computed in step1a, and b represents sensor-B which has been computed in step1b. Then XTR computes $((U_{Sensor-A})^z, g^x, id_{Sensor-B,} id_{Sensor-A,} PK_A)H \oplus b \rightarrow Z_{Sensor-A}$ and $((U_{Sensor-B})^z, g^y, id_{Sensor-B,} id_{Sensor-A,} PK_B)H \oplus a \rightarrow Z_{Sensor-B}$ and XTR sends $(Z_{Sensor-A}, L), (Z_{Sensor-B}, L)$ to Sensor-B.

Step11A: Once Sensor-B receives the sent message, it uses K_B the private key to compute $L^{KB} \rightarrow (U_{Sensor-B})^z$ which is the variable value chosen and computed from the XTR router, and it is used as shard key between the XTR and Sensor-B. As the same time it used K_B to compute $((U_{Sensor-B})^z, g^y, id_{Sensor-B}, id_{Sensor-A}, PK_B)H \oplus Z_{Sensor-B} \rightarrow a$ and sensor-B authenticates with XTR. Now, Sensor-B uses y which is the previously chosen value to compute $a^y \rightarrow g^{xyz} \rightarrow K, (K, id_{Sensor-B}, id_{Sensor-A})H \rightarrow \alpha$ and forwards $(Z_{Sensor-A}, L), \alpha$ to Sensor-A.

Step11B: Sensor-A receives $(Z_{Sensor-A}, L, \alpha)$ and it uses K_A to compute $L^{KA} \rightarrow (U_{Sensor-A})^z$ and $((U_{Sensor-A})^z, g^x, id_{Sensor-B}, id_{Sensor-A}, PK_A)H \oplus Z_{Sensor-A} \rightarrow b$ and authenticates the XTR router. Then, Sensor-A uses x to compute $b^x \rightarrow g^{xyz} \rightarrow K$ and checks whether $(K, id_{Sensor-B}, id_{Sensor-A})H \rightarrow \alpha$ holds or not. If it does not hold, the protocol terminates, Otherwise, Sensor-A is convinced that K is a valid session key. After that, Sensor-A computes $(K, id_{Sensor-B}, id_{Sensor-A})H \rightarrow \beta$ and forwards it to Sensor-B and Sensor-A computes the Session Key $(K, id_{Sensor-B}, id_{Sensor-A})H' \rightarrow Sk_{Sensor-A}$.

Step11C: Upon receiving β, Sensor-B computes $(K, id_{Sensor-B}, id_{Sensor-A})H \rightarrow \beta$ and verifies whether computed β is equal to the received β. If both are equal, then B authenticates Sensor-A and computes the session key $(K, id_{Sensor-B}, id_{Sensor-A}) \rightarrow Sk_{Sensor-B}$

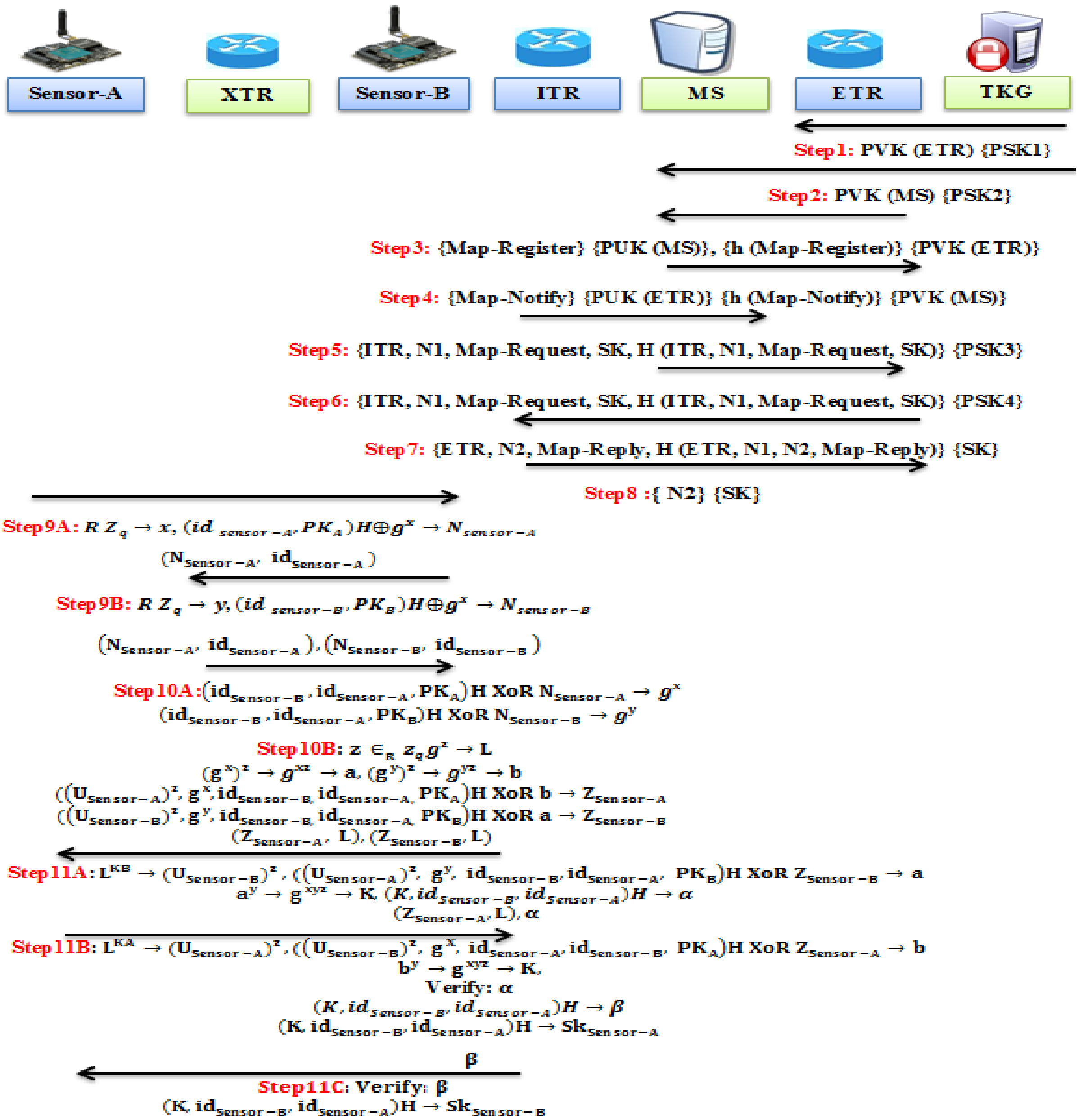


Fig 7.2 Security Refinement Protocol for Internet of things using LISP Network Architecture

7.2.2 Formal Analysis and Results for Refinement Protocols using AVISPA

The proposed refinement protocol has been implemented and evaluated using AVISPA tool. The achieved results have shown that no attack is being found. For this refinement protocol, six basic roles played by Sensor Node device and the LISP network architecture have been defined. K_i and PK_i are private and public keys of Sensor Node; these are stored on the devices used for encryption/decryption and verification of authentication. Here, PSK1 is Pre-Shared key to secure the connection between the TKG and the ETR, PSK2 is the Pre-Shared key to secure the connection between the TKG and MS, PSK3 is the Pre-Shared Key between ITR and MS and PSK4 is the Pre-Shared Key between the MS and ETR. These keys represent the symmetric keys as table 7.2 shows.

Table 7.2 HLPSL Code: Basic Roles

HLPSL Code in AVISPA	**Comments**
role sensor_A (A, B, ITR,ETR : agent,	% This is a role as Sensor_A, with parameters A, B, ITR, and ETR type agent.
PKA : symmetric_key,	%Data-type for symmetric keys of Sensor_A
SND, RCV : channel(dy),	% The RCV and SND parameters are type channel, indicating that the channel type, in this cause (dy), denotes the intruder model to be considered for this channel
H : hash func,	% Data-type for one-way function
G : text)	% it is a finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; both g and q are publicly known
played_by A	% The parameter A appears in the played_by section, which means, intuitively, that A denotes the name of the agent who plays role Sensor_A.
def= *local State : nat*	%Note the local section which declares local variables of Sensor_A
X, Z : text,	% X and Z are a variable value which is chosen and computed from Sensor_A
UA : public_key,	% Data-type for public keys of Sensor_A.
GY , Key, L : message,	% GY,Key, and L % X and Z are a variable value which is chosen and computed from Sensor_A
const sec_m_Key : protocol_id,	%Used to check authentication of Sensor_A
init State := 0	%indicates initialisation of local variables
role sensor_B (A, B, ITR,ETR : agent,	%% This is a role as Sensor_B, with parameters A, B, ITR, and ETR type agent.
PKA, PKB : symmetric_key,	%Data-type for symmetric keys of Sensor_B
SND, RCV : channel(dy),	% The RCV and SND parameters are type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel
H : hash func,	% Data-type for one-way functions
G :text)	%% it is a finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; both g and q are publicly known

played by B	%% The parameter B appears in the played_by section, which means, intuitively, that B denotes the name of the agent who plays role Sensor_B.
def= *local State : nat,*	%Note the local section which declares local variables of Sensor_B
X, Y, Z : text,	%X ,Yand Z are variable value which are chosen and computed from both XTR and Sensor_B
GX, GY : message,	% GX, and GY It is a variable value which is chosen and computed from Sensor_B
UB : public key,	% Data-type for public keys of Sensor_B.
M1: hash(symmetric-key.agent.agent.message.message). message,	%M1 is represented on H(PKA.A.B.exp(G, X).exp(UA, Z)).exp(L, Z),
M2: hash(symmetric-key.agent.agent.message.message). message	% M2 is represented on H(PKB.A.B.exp(G, Y).exp(UB, Z).exp(L, Z)
const sec v Key : protocol id	%Used to check authentication of Sensor_B
init State := 0	%indicates initialisation of local variables
role router_ETR (ETR, MS, TKG,ITR, A,B, : agent	%% This is a role as router_ETR, with parameters ETR, MS, TKG, ITR, A and B type agent.
PKA, PKB: symmetric key	%Data-type for symmetric keys of router_ETR
SND, RCV : channel(dy),	% The RCV and SND parameters are type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel.
H: hash func,	% Data-type for one-way function.
PSK1, PSK2, PSK3, PSK4: symmetric key	%Data-type for symmetric keys of router_ETR
G: text	% It is a finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; both g and q are publicly known
played_by ETR	%The parameter ETR appears in the played_by section, which means, intuitively, that ETR denotes the name of the agent who plays role router_ETR.
def= *local State: nat,*	%Note the local section which declares local variables of router_ETR.
N1, N2, N3,N4: text	%The Nonce is a random number
X, Y, Z: text	% X, Y and Z are variable values which are chosen and computed from both router_ETR
M, M2, ACK: messages	% acknowledgment
UA, UB, PK_ETR, PK_MS: Public key	% Data-type for public keys of router_ETR..
K_ETR, K_MS: {text.public_key}_ inv(public_key),	%Compute keys question by router_ETR
SK: symmetric	%Data-type for symmetric keys (session Key) of router_ETR
EIDPre: messages	%Endpoint Identifiers
GY, Key, L: message,	% GY, Key and L are variable values which are chosen and computed from router_ETR
init State := 0	%indicates initialisation of local variables
role router_ITR (ETR, MS, A,B,TKG : agent,	%% This is a role as router_ITR, with parameters ETR, MS, TKG, ITR, A and B type agent.
PKA, PKB: symmetric key	%Data-type for symmetric keys of router_ITR
SND, RCV : channel(dy),	% The RCV and SND parameters are type channels, indicating that the channel type, in this cause (dy), denotes the intruder model to be considered for this channel.

H: hash func,	% Data-type for one-way functions.
PSK1,PSK2,PSK3,PSK4 : symmetric key	%Data-type for symmetric keys of router_ITR
G: text	% It is a finite cyclic group $\mathcal{G}$ generated by an element g of prime order p; both g and q are publicly known
played_by	% The parameter ITR appears in the played_by section, which means, intuitively, that ITR denotes the name of the agent who plays role router_ITR.
def= *local State: nat,*	%Note the local section which declares local variables of router_ITR.
N3,N4: text	%The Nonce is a random number
X, Y, Z :text	% X, Y and Z are variable values which are chosen and computed from both router_ITR
M,M2,ACK :messages	% acknowledgment
UA, UB, PK_ETR,PK_MS: Public key	% Data-type for public keys of router_ITR..
K_ETR, K_MS: {text.public_key}_ inv(public_key),	%Compute keys question by router_ITR
SK: symmetric	%Data-type for symmetric keys (session Key) of router_ITR
EIDPre: messages	%Endpoint Identifiers
GY, Key, L: message,	% GY, Key and L are variable values which are chosen and computed from router_ETR
init State := 0	%indicates initialisation of local variables
role map_server (MS,ETR,ITR: agent,	% This is a role as map_server, with parameters, MS, ETR, and ITR type agent.
PSK1,PSK2,PSK3,PSK4 : symmetric key,	%Data-type for symmetric keys of map_server
SND ,RCV : channel(dy),	% The RCV and SND parameters is type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel.
H: hash func,	% Data-type for one-way functions.
played_by MS	% The parameter MS appears in the played_by section, which means, intuitively, that MS denotes the name of the agent who plays role map_server.
def= *local State: nat*	%Note the local section which declares local variables of map_server.
N1, N2,N3,N4: text	%The Nonce is a random number
M,M2,ACK :messages	% acknowledgment
PK_ETR,PK_MS: Public key	% Data-type for public keys of map_server.
K_ETR, K_MS: {text.public_key}_ inv(public_key),	%Compute keys question by map_server.
SK: symmetric	%Data-type for symmetric keys (session Key) of router_ETR
init State := 0	%indicates initialisation of local variables
role the_trusted_ticket_granting (TKG,ETR,MS: agent,	% This is a role the_trusted_ticket_granting with parameters TKG ETR and MS type agent.
PSK1,PSK2: symmetric_key,	%Data-type for symmetric keys of the_trusted_ticket_granting.
PUK_ETR , PUK_MS: Public_Key	% Data-type for public keys of the_trusted_ticket_granting.
H : hash_func,	% Data-type for one-way functions.
Snd, Rcv : channel(dy),	% The RCV and SND parameters is type channel, indicating that the channel type, in this case (dy), denotes the intruder model to be considered for this channel.

played_by TKG	% The parameter TKG appears in the played_by section, which means, intuitively, that TKG denotes the name of the agent who plays role the_trusted_ticket_granting.
local State : nat,	%Note the local section which declares local variables of the_trusted_ticket_granting.
N1 : text	%The Nonce is a random number
M,M2,ACK, : messages	% acknowledgment
PVK_ETR,PVK_MS:{text.public_key}_ inv(public_key)	%Compute keys question by the_trusted_ticket_granting.
EIDPre:messages	%Endpoint Identifiers
init State := 0	%indicates initialisation of local variables

After defining the #basic roles, it is essentially needed to define the composed roles which describe the sessions of the protocol. The # Transitions represent the receipt of messages and the sending of reply messages, i.e., showing all the messages exchanges while the input code and results are described in the appendix-G. The #composed roles have no transition section, but rather a composition section in which the basic roles are instantiated. The /\ operator indicates that these roles should execute in parallel. In the *session role*, it usually declares all the channels used by the basic roles. These variables are not instantiated with concrete constants. The *channel* type takes an additional attribute, in parentheses, which specifies the intruder model that is assumed for that channel. Here, the type of the declaration *channel (dy)* stands for the Dolev-Yao intruder model [AVISPA, 2013]. Under this model, the intruder has full control over the network, i.e., all messages sent by agents will go to the intruder. He may intercept, analyze and /or modify message (as far as he knows the required keys) and send any message he composes to whoever he pleases, posing as any other agents. Finally, a top-level role is always defined. This role contains global constants and a composition of one or more sessions, the Table 7.3 below shows the #intruder may play some roles as a legitimate user. There is also a statement which describes what knowledge the intruder initially has. Typically, this includes the names of all agents, all the symmetric keys and any shares with others. Note that the constant ‘I’ is used to refer to the intruder, as the source code below.

Table 7.3 HLPSL Code: Intruder Information Heading

HLPSL Code in AVISPA	**Comments**
intruder knowledge = {a, b, itr,etr,ms,tkg, g, pi, ua, ub, ui ,pk_etr,pk_ms,k_etr,k_ms, h,psk1,psk2,psk3,psk4,m,m2 sk,maprequest, mapreply}	% Specifies the intruder’s knowledge and capabilities. Check the intruder between the Sensor-A, Sensor-B, ETR, ITR and MS. And also check public keys UA and UB and the pre-shared key PSK1,PSK2,PSK,3 and PSK4 if intruder captures between the sensor-A and Sensor-B, and ITR,ETR and MS.
Composition	% It is one or more basic roles, gluing them together so they execute together, usually in parallel with interleaving semantics.

session(b, a, itr,etr h, pa, pb, ua, ub, g)	% In the session, one usually declares all the channels used by the basic roles. Likewise, set session of Sensor-A, ITR, and ETR defined the Public keys UA, and UB on the network environment.
/\ session(i, b,etr, itr, h, pi, pb, ui, ub, g) */\ session(a, i, itr,etr, h, pa, pi, ua, ui, g)* */\ session(etr,i,k_etr,h)* */\ session(ms,i,k_ms,h)* */\ session(itr,ms,etr, psk3,psk4,sk,h)* */\ session(ms,i,psk3,psk4,sk)* */\ session(itr,i,sk,h)* */\ session(etr,i,sk,h)*	% The /\ operator indicates that these roles should execute in parallel. In the session, one usually declares all the channels used by the basic roles. Also, set session of intruder checker on the network environment.
end role	%End of session role

Table 7.4 shows the #Specifying Security Goals are specified in HLPSL by augmenting the transitions of the basic roles with the so-called goal facts and by then assigning them a meaning by describing, in the HLPSL *goal* section, what conditions –i.e., what combination of such facts indicate an attack and a violation of *secrecy*. The goal declaration section describes that it should be considered as an attack when the intruder learns a secret value internally; the attack conditions are specified in terms of temporal logic, but useful and concise macros are provided for two most frequently used security goals, authentication and secrecy.

Table 7.4 HLPSL Code: Specifying Security Goals

HLPSL Code in AVISPA	**Comments**
Goal	% Specifies the security properties to be cheeked
securecy_of Map_server, Router_ETR Router_ITR Sensor_A,Sensor_B on psk1,psk2,psk3,psk4,n1,n2,n3	% Check the secrecy of random nonce N1,N2,N3 and pre-shared key PSK1,PSK2,PSK3, and PSK3 between MS,ETR, and ITR and between the Sensor-A and Sensor-B
Router_ETR weak authenticates Server_MS	% Check the authentication between ETR and MS, and then ETR should provide confirmation witness information to MS.
Router_ITR weak authenticates Map_Server	% Check the authentication between ITR and MS, and the ITR should provide confirmation witness information to MS.
Router_ITR weak authenticates Router_ETR	% Check the authentication between ITR and ETR, and the ITR should provide confirmation witness information to ETR.
Router_ETR authenticates Server_MS on n1 authentication on_n1	% strong authentication between ETR and MS on value of random nonce N1.
Router _ETR authenticates Router _ITR on n2 authentication_on n2	% strong authentication between ETR and ITR on value of random nonce N2.
Router _ITR authenticates Router _ETR on n3 authentication_on n3	% strong authentication between ITR and ETR on value of random nonce N3.
sensor_a authenticates aensor_b on sk	% strong authentication between Sensor-A and Sensor-B on session key SK.
End goal	%End of session goal

7.2.3 Analysis of Results

In this section the refinement protocol tests by Automated Validation of Internet Security Protocols and Applications (AVISPAP). It has been used four back-end tools which is integrated by AVISPA, namely; the On-the-fly Model-Checker OFMC, the Constraint-Logic-based Attack Searcher CL-AtSe, the SAT-based Model-Checker SATMC, and the TA4SP protocol analyser, which verifies protocols by implementing tree automata based on automatic approximations. All the back-ends of the tool analyse protocols under the assumptions of perfect cryptography and that the protocol messages are exchanged over a network that is under the control of a Dolev-Yao intruder [AVISPA, 2013]. That is, the back-ends analyse protocols by considering the standard protocol independent, asynchronous model of an active intruder who controls the network but cannot break cryptography; in particular, the intruder can intercept messages and analyse them if he possesses the corresponding keys for decryption, and he can generate messages from his knowledge and send them under any party name as figure 7.3 shows.

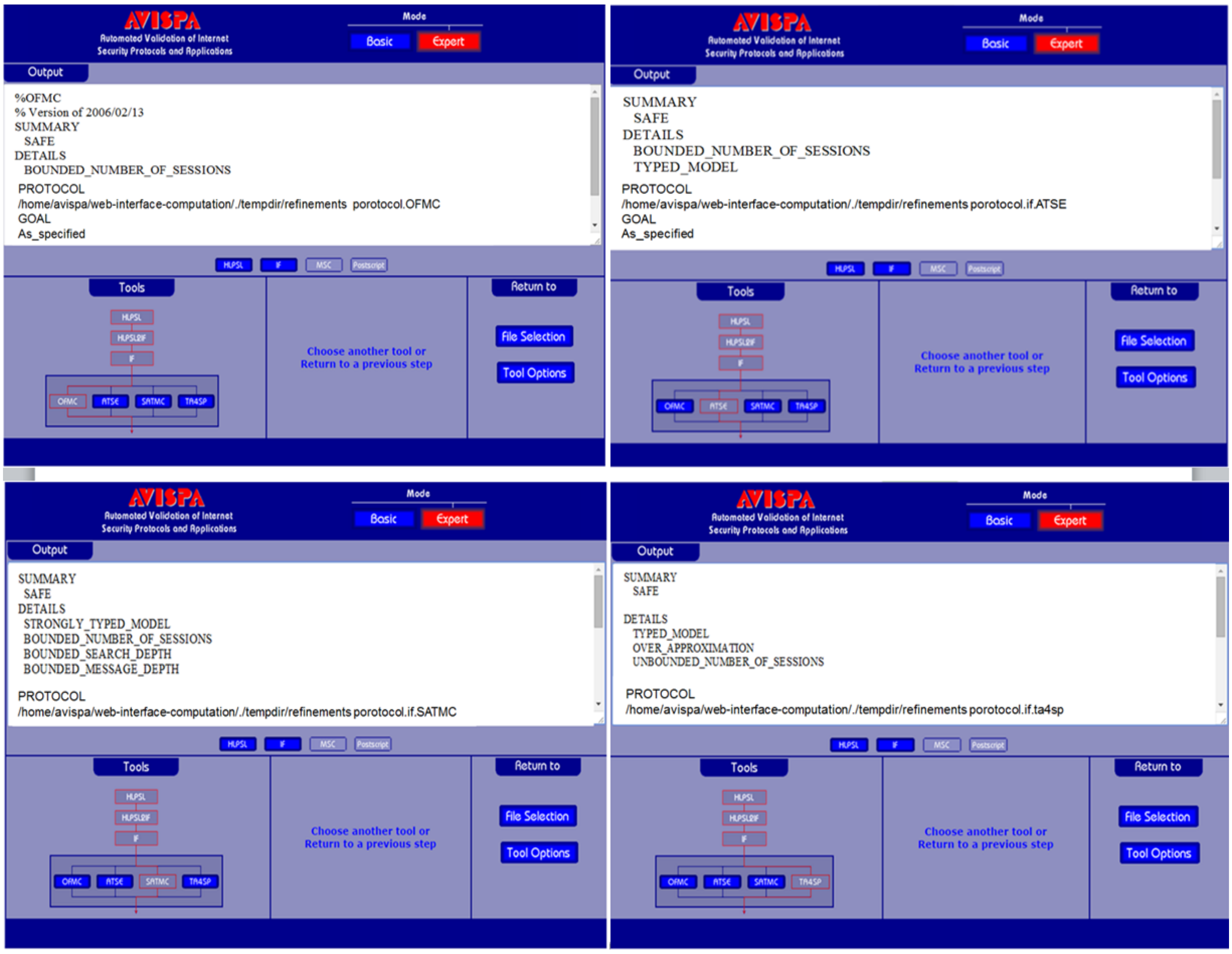


Fig 7.3 AVISPA Results

Table 7.5 shows the results of the security refinement protocol for IoT based on LISP architecture, while the input code and results are described in the appendix section -G. Also, AVISPA is used to verify the security properties like secrecy, integrity and authentication. It gives details about whether the protocol is safe or not. If not it gives then the trace of the attacks found, to indicate secrecy attack or authentication attack. Consequently even through many properties of the protocol are to be checked, only few can be verified using AVISPA. For this the modelling is done in HLPSL, language used by AVISPA tool. For our verification, OFMC, ATSE, SATMC, and TA4SP have been used to search for the attacks on the protocol. The feature of finding and tracing the attack makes AVISPA different from other tools. Henceforth, the tool is tested and the protocol verification results are analysed; they show that they do not have any security flaws and the security refinement protocol is safe to be used.

Table 7.5 AVISPA Tools (OFMC, ATSE, SATMC, and TA4SP) Results

Version	Tool	Description	Result
Basic session	OFMC	Visited Nodes: 4765 nodes Depth: 6 plies Search Time: 0.6s	**SAFE**
Basic session	ATSE	Analysed: 6844 States Reachable: 22735States Translation: 0.00 seconds Computation: 0.01 seconds	**SAFE & goal as specified**
Basic session	SATMC	STATISTICS Attack Found : false Boolean Upper Bound Reached: true Boolean Graph Leveled off: 4 steps Sat Solver: zchaff Solver Max Steps Number: 11 Steps Steps Number : 18 Steps Atoms Number: 423 Atoms Clauses Number 1412 Clauses Encoding Time: 0.1 Seconds If2Sate Compilation Time 0.01 Seconds ATTACK TRACE %%no attacks have been found…	**SAFE**
Basic session	TA4SP	STALISTICS SECURITY-As specified ATTACK TRACE No attack found	**SAFE**

7.3 Simulation and Performance Analysis

As a beginning, IP-WSN consists of a large number of small size sensor nodes deployed in the observed environment. As shown earlier, Sensor nodes have a small memory (8K of total memory and disk space) and a limited computation power (8-bit, 4 MHZ CPU) [Singh et al., 2017]. They usually communicate with a power base station, i.e. X Tunnel Routing (XTR) which connects sensor nodes with external network, e.g. ID/Locator Spilt Protocol (LISP). The limited energy at sensor nodes creates some hindrances in implementing complex security schemes. There are two major factors for energy consumption. The First factor is the transmission and reception of data while the second is the processing of query request.

Consequently, the sensor network environment simulator is built in Contiki and Cooja tool. The network simulated consists of 100 nodes. The sink node (XTR) is placed at the corner to maximise network depth. Connectivity information is derived from empirical data collected from a real world study. Furthermore, the simulation tool and the results show that the refinement protocol memory is efficient as it requires 72 bytes of memory storage for keys, where transmission and reception cost per connection is 75.125 bytes.

In order to demonstrate the performance evaluation of security refinement protocol, a randomly 100 sensor nodes plus one XTR in 1000 by 1000 terrain has been simulated. Basic simulation parameters employed are described in table 7.6.

Table 7.6 Simulation Parameters in Cooja

Terrain	1000x1000
Total Number of Nodes	101 (including XTR)
Initial battery of each sensor node	$1x10^6$J
Power consumption for transmission	1.7 W
Power consumption for reception	1.3 W
Idle power consumption	1.16 W
Carrier sense threshold	3.631E- 10W
Receive power threshold	1.557e – 11 W
Frequency	9.15e8
Transmitting & Receiving antenna gain	2.0

The figure 7.4 shows the security refinement protocol designed in a network layer at sensor node architecture which is simulated in Cooja tool.

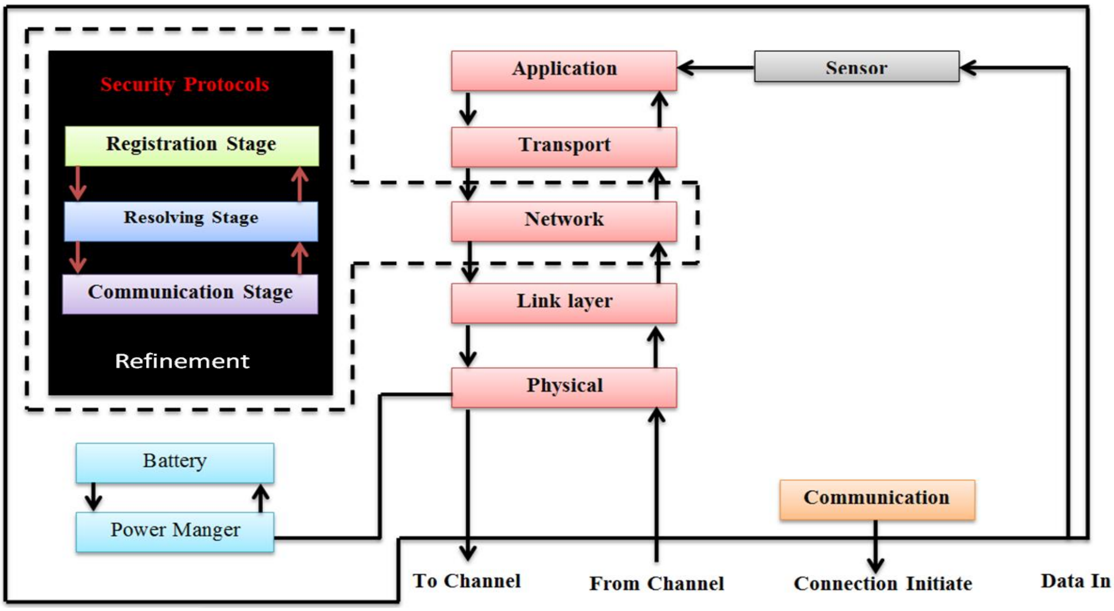


Fig 7.4 Integrate the Security Protocols in Sensor Node Architecture

The figure 7.5 shows the WSN network simulated consists of 100 nodes, where is built in Contiki and Cooja tool.

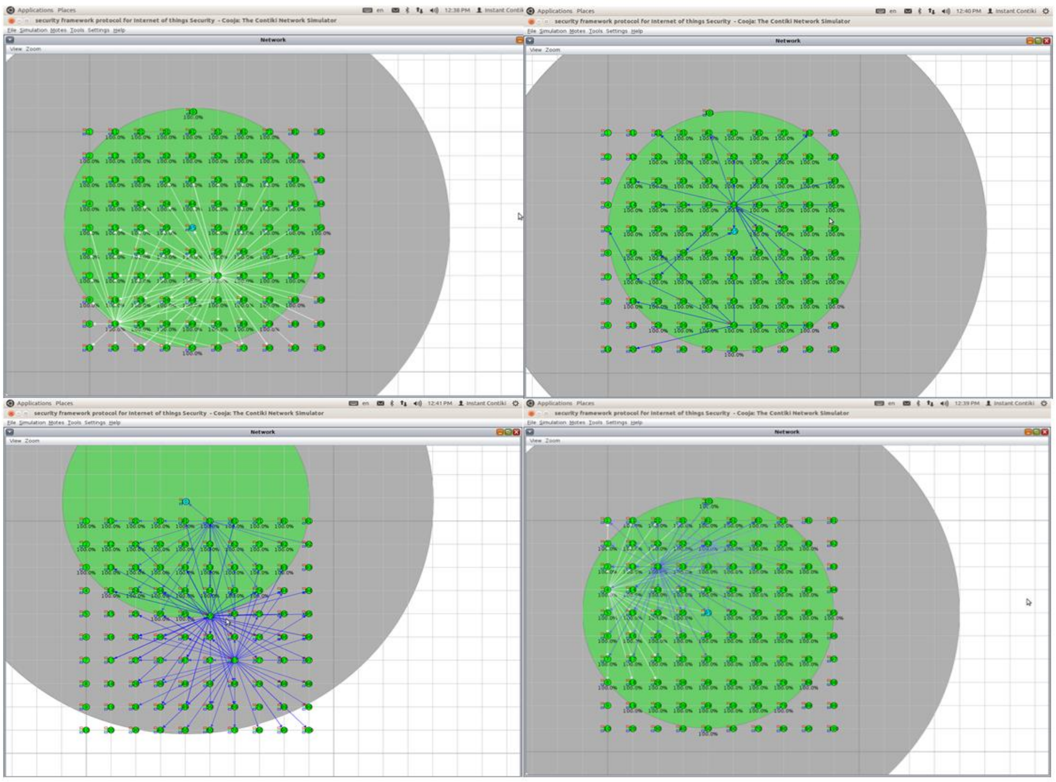


Fig 7.5 Nodes Simulation in Cooja tool

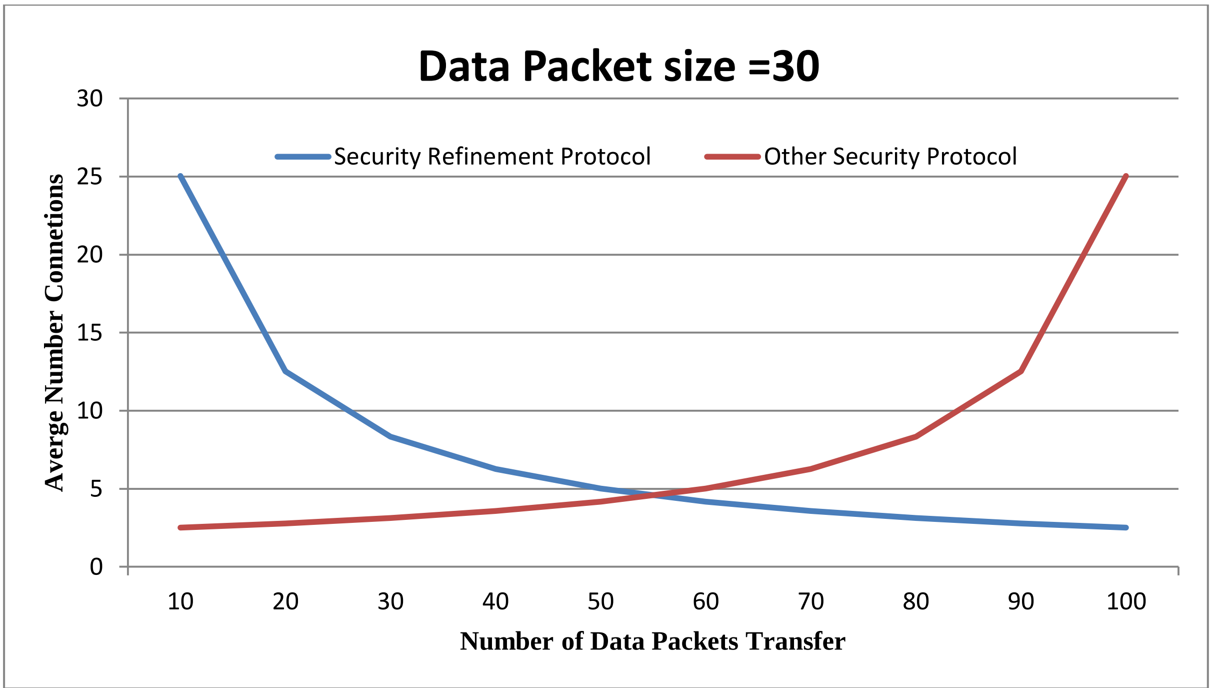
7.4 Performance Analysis of Communication Overhead

In the simulation scenario in cooja, the application sent data packets of size 30 bytes in a periodic interval. The communication overhead of Security refinement protocol for one communication is decreased in the transference of a number of data packets as shown in 8.8 .

Communication Overhead (CO %) is calculated as:

$$\textbf{Communication Overhead }(\mathbf{CO\%}) = \left(\frac{\mathbf{Tn*75.125}}{\sum_{\mathbf{i=0}}^{\mathbf{n}} \mathbf{N_i^p * 30}}\right) * \mathbf{100} \qquad (\mathbf{1})$$

Here , ‘Tn’is the total number of connections and $N_i^p$is the number of packets transferred by node i. whereas,75.125 bytes have been multiplied to Tn because every connection security refinement exchanges messages or packets during the authentication and key exchange phase whose cumulative size is 75.125 byte. The size of each data packet is 30 bytes.



7.6. Communication Overhead (%) of Security Refinement Protocol to Internet of Things

Figure 8.8 shows the relationship between the number of data packets transferred and the average number of connections. Likewise it shows the comparison of this research study security protocol proposed with other security protocols. As seen from the chart above, the communication overheard decreases from 25 blue lines if the number of the connections to the network is less. On the other hand, if the number of connections increases in the red line the communication overhead increases as well.

7.5 Performance Analysis of Power Computation for Security Refinement protocol

Power Computation depends primarily on which type of protocol is used whether symmetric or asymmetric. The computation power required for symmetric encryption and decryption scheme is assumed to be CSEN and CSDN respectively and computation power of asymmetric encryption and decryption is CAEN and CADN respectively too. Then, the total power consumption required by single node during first two phases is

$$\boldsymbol{Power\ Computation = (CSEN + CSDN) + (CAEN + CADN)} \qquad (\mathbf{2})$$

Computation power required by a single node during data transmission phase is calcite as,

$$\boldsymbol{Power\ Computation = (TNSDP * CSEN) + (TNRDP * CADN)} \qquad (\mathbf{3})$$

Where the Total Number of Sent Data Packets is (TNSDP) and the Total Number of Received Data Packets is (TNRDP).

7.6 Performance Analysis Memory Consumption in the IP-WSN

Every IP-WSN node needs to store only six keys, three of them are permanent whereas the other three are temporary. Permanent keys consist of one public key and one private key; the public key shares with XTR. Temporary keys consist of a public key, a shared secret key and of another node and session secret keys. In order to save these keys, only 72 bytes are needed. This approach will make sensor network memory efficient. Details of this approach are given in table 7.7.

Table 7.7 Storage Requirement of Keys in Sensor Node Device

Store Number	**Keys**	**Size (in bytes)**
Permanent Keys		
1	Public Key of Sensor Node	16 byte
2	Private Key of Sensor Node	16 byte
3	Shared Secret key with XTR	8 byte
Temporary Keys		
4	Public key of other Node	16 byte
5	Shared Secret key of other Node	8 byte
6	Session Key	8 byte
Total Storage Size Required		72 bytes

7.7 Performance Analysis of Energy Consumption in the IP-WSN

The main source of energy consumption at IP-WSN node is its transmission and reception cost. Cooja simulation is used to calculate the consumption energy of the sensor node in different modes: Transmit Receive, Idle and Sleep. Energy consumption rate of each node is given in Table 7.7. For each connection, the security

control packets of cumulative 75.125 bytes require authentication and key exchange mechanism, which is an acceptable trade-off between energy and security. The simulation result of energy consumption is shown in figure 7.7. When the pulse of the energy increases as shown in section 2, it means that the device is in an active mode, but when the pulse drops down as section 1 shows, it means that the device is in the sleep mode. The rectum and the winding line in section 3 mean that the device is in the idle mode.

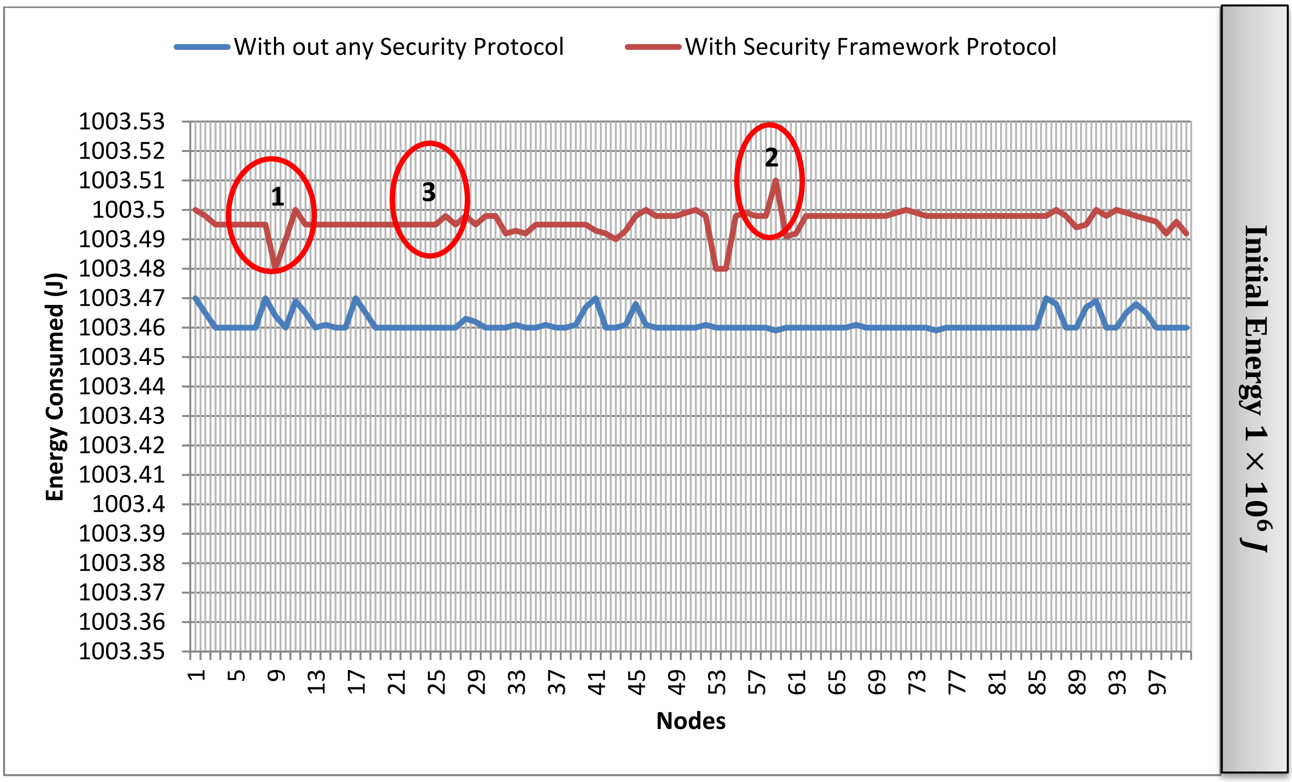


Fig 7.7 Energy Consumptions for IP-WSN

7.8 Resilience against Node Compromise

This means that the single node compromise will not expose the whole communication in the network. In other words, only the communication links that are established with compromised node can expose the network. Let's Suppose 'SNCN' is the set of sensor node that establishes the connection and 'SNCNP' is the set of compromised nodes. Then, SNCN

∩ SNCNP will give a set of nodes that are compromised as well as connected. The maximum number of connections in equation (4) can be exposed only if all compromised nodes are connected to uncompromised nodes. On the other hand, the minimum numbers of links in equation (5) can be exposed only if all compromised nodes are connected with each other.

$$\boldsymbol{Maximum\ Compromised\ Linkes}: \boldsymbol{SNCN\ \cap SNCNP} \qquad (\mathbf{4})$$

$$\boldsymbol{Minimum\ Compromised\ Links} \begin{bmatrix} \frac{\boldsymbol{SNCN \cap SNCNP}}{\mathbf{2}} \boldsymbol{for\ \rightarrow even} \\ \frac{\boldsymbol{SNCN \cap SNCNP+1}}{\mathbf{2}}\ \boldsymbol{for \rightarrow odd} \end{bmatrix} \qquad (\mathbf{5})$$

If IP-WSN network is assumed to consist of 1000 nodes and a total of 500 connections are established between a pair of nodes, the total links that can be minimum and maximum compromise will be as shown in figure 7.8. If the numbers of compromised links increase the maximum connection of compromised nodes rises up as the red line in the figure shows. On the contrary, if the numbers of the compromised link decrease, the minimum of the compromised nodes decrease as shown by the blue line.

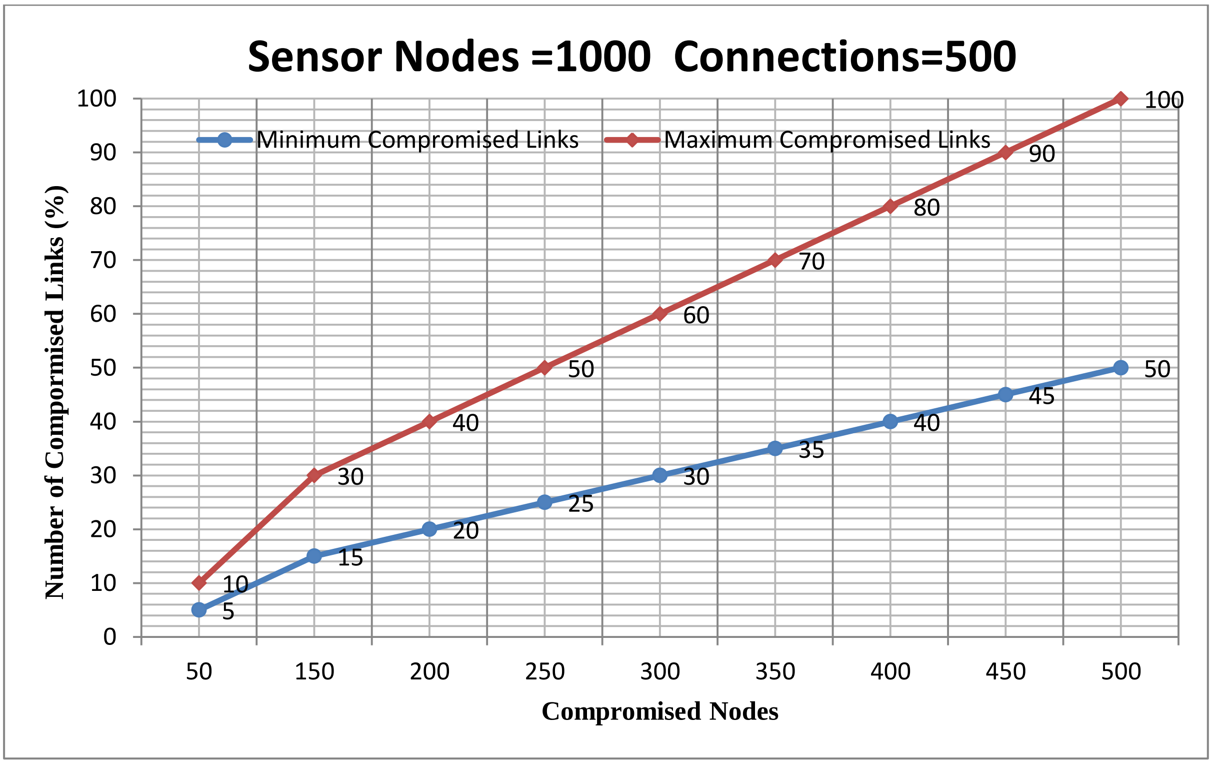


Fig 7.8 Percentage of Compromised Links (IP-WSN)

7.9 Summary

In this chapter, a security refinement protocol has been proposed to the IoT using LISP network architecture. Furthermore, the refinement is based on three security protocols namely resignation, resolving and communication which have been already introduced in previous chapters. Noticeably, the main idea of these protocols is to set an interface for each phase of the protocol to be joined together as one refinement protocol. Moreover, the security refinement protocol analysis and verification using AVISPA show that no attacks have been detected. Finally, simulating the security refinement protocol in Contiki and Cooja simulation tool and the results show that the security refinement is highly scalable and the memory is quite efficient. It only needs 72 bytes of memory to storage the keys in the device and it introduces 75.125 bytes of transmission and reception cost per connection. In this context, it has the advantage of securing defines against compromised nodes.

Consequently, chapter 8 a comparison of the recent security protocol for IoT against the present research study protocols

Chapter 8:

A comparison of Security Protocols for Internet of Things

8.1 Introduction

The Internet of Things (IoT) has recently become an important research topic because it integrates various sensors and objects to communicate directly with one another without human interference. Furthermore, the requirements for the large scale deployment of the IoT are rapidly increasing with a major security concern. Therefore, a new security approach for IoT communications proposed in chapter 7 provided End-to-End Security (E2E) communications to IoT using Locator/ID Separation Protocol (LISP). Consequently, this chapter provided a comparison of security protocols for IoT against the present security protocol of this research. Accordingly, the structure of this chapter is divided as the following: section 8.2 discusses the secure IoT. Section 8.3 discusses the recent security protocols for the IoT. Finally, a summary concludes the chapter in section 8.4.

8.2 Secure Internet of Things

Internet Protocol Version 6 (IPv6) [Nicolls et al., 2017] offers interconnection of almost every object with the Internet. Furthermore, it leads to massive possibilities to develop new applications for the IoT, such as home automation and home security management, smart energy monitoring and management, item and shipment tracking, smart cities and health monitoring. Therefore, due to the global connectivity and sensitivity of applications section 9.2.1, provided the main security requirements which are necessary in IoT devices [Yang et al., 2017], and are discussed as well.

8.2.1 Security Requirements in Internet of Things

Confidentiality: Messages that flow between a source and a destination could be easily intercepted by an attacker and secret contents are revealed as a result. Consequently, these messages should be hidden from the intermediate entities; in other words, End-to-End (E2E) message secrecy is required in the IoT.

Data Integrity: No intermediary between a source and a destination should be able to change secret contents of messages without being discovered, e.g. a medical data of a patient. Furthermore, stored data should not be modified without being noticed. Message Integrity Code (MIC) is mostly used to provide this service.

Authentication: Communicating end points should be able to verify the identities of each other to ensure that they are communicating with the entities who they claim to be.

Availability: For smooth working of the IoTs and access to data whenever needed, it is also important that services offered by applications should be always available and work properly. In other words, intrusions and malicious activities should be detected. Intrusion Detection Systems (IDSs) and firewalls, in addition to the security mechanisms above, are used to ensure the availability of the security services.

Freshness*:* Considering both data and key to ensure that there is no replays of old messages because unsecured router and IoT node causes unnecessary old replays.

Non-Repudiation: Last but not least, a compromised intermediate node can store a data packet and replay it at a later stage. The replayed packet can contain a typical sensor reading (e.g. a temperature reading) or a paid service request. It is, therefore, important that there should be mechanisms to detect duplicate or replayed messages. Replay protection or freshness security services provide this; they can be achieved through integrity-protected timestamps, sequence numbers, nonces.

Resiliency: It provides an acceptable level security even if some IoT nodes are compromised.

8.3 Security Protocols for the Internet of Things

This section discusses different security protocols which have been proposed to the IoT devices.

[Figueroa et al., 2012] has proposed a lightweight security protocol to access web services in Low-power Wireless Personal Area Networks over IPv6 (6LoWPAN). The protocol's objective is to provide a reliable end to end security communication for IoT/6LoWPAN by using a compression and decompression of Internet protocol. Furthermore, the protocol provides confidentiality to IoT networks via the use of SNOW Stream cipher. However, this protocol does not address a number of attacks. For example, if the adversary captures one of the sensor nodes of IoT/6LoWPAN, he can find out about the cryptographic data which is stored in the sensor node and discloses the network confidentiality. Adding to this, the attacker can launch DoS and wormhole/ sinkhole attacks that make the sensor nodes believe that they are a neighbour node and forward the packets between them. This may cause confusion to the getaway to locate the node or receive false data. Moreover, by launching a rushing attack through the deployment nodes, this could cause the breakdown of the communication between the source and the destination by transmitting a huge number of packets at the same time.

[Zhou et al., 2011] has proposed an amended security gateway based on 6LoWPAN, which connects IoT/6LoWPAN with IPv6 network. The proposed protocol used an SNEP mechanism to achieve authentication and confidentiality through providing a secure guarantee for communication between networks. The main objective of this protocol is to provide security between the getaway and the node against the malicious nodes or any suspect attacks that can compromise the network. However, this protocol does not address the resource consumption attacks i.e. replayed attacks, DoS and physical node capture attacks. E.g. the adversary captures a sensor node (6LoWPAN) via using selective forward attacks/or even stealing cryptographic material which is stored on the node by injecting fake packets in the networks. The attacker can launch a man in middle attack between the getaway and the sensor node and steal/or modify the information between them. Furthermore, Sybil attacks, where can deliver false information message to the getaway through malicious nodes.

[Khan et al., 2012] has proposed an authentication and mutual key establishment scheme for IP based wireless sensor network (6LoWPAN). The authentication has been achieved in 6LoWPAN via Eliptic Cure Cryptosystem (ECC). Although, public key cryptography is costly in terms of WSN (6LoWPAN) as shown in [Yang et al., 2017], [Masdari et al., 2017] and [Mstaf et al., 2017]. Furthermore, the authors considered virtual network architecture which combines two sub networks connected to each other through edge routers. The specific node in the network acts as a reference for the different supported security functionalities, and is called the Network Security Manager. They stated in their work that the node should be authenticated with each other by generating the authentication keys to secure the communication link which are the public/ private keys for encryption and decryption of the message. Therefore, a node should process four steps in order to achieve successful authentication with key establishments; firstly, each entity of the network generates a random number which is assigned by the Network Security Manager after a node registration phase. Secondly, one entity of network shares the assigned public key, while others share the public key has generated by the relevant local Network Security Manager. Thirdly, the Author considered the secure communication between two nodes in the network which are the source IP and the destination IP that are used to generate a specific elliptic curve. Fourthly, each entity and network has its own generator G_e and G_n respectively. However, if the adversary compromises nodes, he can launch a combination of the wormhole and sinkhole attacks which can affect the confidential data of the (6LoWPAN).

[Jara et al., 2011] has proposed a security protocol for 6LoWPAN based on the ID/Locator split architecture which is an extension of the Return Reputability (RR) process with ECC-based asymmetric cryptography in order to carry out scalable inter-domain authentication. Basically, this protocol is designed to stop any malicious nodes from establishing false updates of the system location, and also to prevent some packets from reaching their intended destination diverting some traffic to the intruder/or flooding third parties with unwanted traffic. Therefore, the main goal of this protocol is to allow authentication for the mobile nodes in the visited network; besides providing authentication to the getaway in order to achieve a secure mobility management of the mobile nodes (6LoWPAN). However, the author used RR security which depends mainly on the Internet to ensure the IP address. Since RR is based on the communication between two entities; it can be more susceptible to being spoofed by any attacker. Referring to the registration stage of the node, the adversary can launch a man in middle attack between 6LoWPAN nodes and the getaway and can steal the cryptographic data through the exchange of the transactions update, then he can modify/or corrupt the data by sending false information to the getaway or even to the victim node.

[Kothmayr et al., 2013] has proposed a security authentication protocol for 6LoWPAN based on RSA mechanism which uses public key cryptography algorithm. The objective of this protocol is to perform authentication in the Datagram Transport Layer Security (DTLS) between nodes and the source publisher via the use of handshake which is based on an exchange of x.509 certificates containing RSA keys. Furthermore, the security protocol provides message integrity, confidentiality and authenticity. However, the author did not consider the encryption of data between the 6LoWPAN nodes. Therefore, a malicious node can spoof on the original node information which can cause confusion to the system by transmitting false data.Even more it can claim to be an original node to the getaway/or other neighbouring nodes by using act technique.

[Ikram et al., 2009] has proposed a simple lightweight authentic bootstrapping protocol for IPv6 based on 6LoWPAN by using AES encryption which is an encryption standard in IEEE 802.15.4. The purpose of this protocol is to provide resource efficiency and security features assured by securing the communication between nodes. Furthermore, this protocol depends on the key management infrastructure and addresses different types of attacks such as, replay attack, location privacy attack, passive eavesdropping, DoS attack and data loss attack. The author assumed that every node (RFDs and FFD) in the 6LoWPAN is equipped with AES-CMAC-128, AES-CTR and AES- CCM-128. However, the adversary can launch an overwhelming attack, which can destroy the routing by generating a lot of traffic in order to affect the performance of the getaway. Moreover, if the adversary compromises nodes; he can launch a combination of wormhole and sinkhole attacks in order to manipulate the use of the routing lists that are included in the route request query. Adding to this, an adversary can manipulate the end-to-end integrity control by modifying a number of messages which will have to travel to their destination to discover that they have been altered. This means that the energy is wasted due to the fact that integrity violations are not detected as soon as possible and the maliciously modified packet is still forwarding to its destination.

[Brachmann et al., 2012] has proposed end to end transport security in the IP-based Internet of Things via using different scenarios i.e. HTTP/CoAP, and TLS/DTLS. This protocol provides E2E security between two devices located in homogeneous networks using either HTTP/TLS or CAP/DTLS by proposing a mapping between TLS and DTLS. However, CoAP does not itself provide protocol primitive for authentication or data encryption. Therefore this protocol does not address flooding replay and amplification attacks. There is no authentication to identify 6LoWPAN devices claim. This protocol just provides E2E security between the node and the getaway, so if the adversary captures a node and gets the global security information before the security setup is finished; he can obtain the security information within its vicinity.

[Raza et al., 2013] has proposed a lightweight protocol security CoAP for the Internet of Things. The author provided an investigation to reduce the overhead of DTLS in 6LoWPAN header compression by integrating DTLS and CoAP for Internet of Things. However, the proposed protocol provided communication security (End-to-End Security) to the 6LoWPAN devices by comparing the DTLS. However, it does not address the authentication or the encryption process, so any malicious node can claim that it is the original node and can communicate with the getaway or even act as fake getaway and steal all the nodes' information. Furthermore, spoofing on the data can occur easily here since there is no encryption procedure to provide confidentiality. Therefore, the attacker can track the legitimate encrypted packet of the node. It can copy the encrypted data from the node and give false information to the getaway. Denial of Service attack (DoS) can attack 6LoWPAN devices that use lightweight IPsec which can cause extra load on the network and breakdown the communication link.

[Kim, 2008] has provided analysis for security threats to the 6LoWPAN adaptation layer from the point of view of IP packet fragmentation attacks. The proposed work showed that IP fragmentation is the attack that can most affect the 6LoWPAN. A security mechanism against the packet fragmentation attacks and replay attacks has been proposed. This security mechanism uses Timestamp and None Options that are added to the fragmented packets at the 6LoWPAN adaptation layer. Nevertheless, the mechanism does not address a number of attacks e.g. Packet drop attack/or blackhole which can occur when the router is compromised due to different causes; one of these causes is occurs through the DoS attack because packets are routinely dropped from a network. The adversary can effectively launch a combined rushing and wormhole attack during the neighbour discovery phase and convince the remote sensor nodes that he is one of the neighbouring nodes and adding him to their list.

[Bonetto et al., 2012] has investigated the ability to secure the communication for smart IoT objects. The objective of this work is to design a lightweight protocol procedure to set up secure end to end channels between unconstrained and remote peers and IoT devices. The author addressed security in terms of resilience against node capture via using lightweight IPsec security association. However, this is not enough to provide a high level of protection to the network. The adversary can launch a DoS attack which can affect the performance of the network. Also, the attacker can capture legitimate nodes by launching the selective forwarding attack or by combining the wormhole/ sinkhole/ rushing attacks which affects the communication between node and getaway.

[Raza et al., 2011] has proposed lightweight IPsec protocol to secure the communication between sensor nodes in 6LoWPAN and the hosts in the IPv6-enabld Internet. The goal of this protocol is to provide end to end security via using existing methods and infrastructures. Also, it provides confidentiality and data integrity between the sensor node and the 6LoWPAN router which is connected to the Internet source. However, the attacker can sniff the legitimate encrypted packet of the node. It can copy the encrypted data from the node and give false information to the getaway. DoS attacks can occur in 6LoWPAN devices that use lightweight IPsec protocol which can cause an overloaded network and breakdown the communication link.

[Jung et al., 2009] has proposed a lightweight protocol for IP-WSN (6LoWPAN) via using ECC based lightweight SSL. The objective of this protocol is to secure both sensors and clients who are connected to the Internet, and that has been achieved by using ECC and SSL which is based on the handshake protocol. The handshake protocol allows the sensor and getaway which are connected to the Internet to be authenticated by negotiating cryptographic algorithms and keys. The author has used ECC 160 bit which provides the same level of security as RSA using 1024 bit key according to [Mstafa et al. 2017]. This study has proved that ECC 160 exchange key operations are 13 times faster than 1024 bit RSA decryption operations on the mode. The protocol provided authentication and confidentiality. Although there is end to end security between the WSN and getaway which is connected to the source (internet); the adversary can launch Man in middle attacks which can set between the WSN (6LoWPAN) and the getaway as a third party and spoof on the data or even modify and send it to other nodes or getaways.

8.4 Summary

In this chapter provided a security analysis protocols to IoT devices. The outcome of this analysis shows the recent design for the security protocols does not reached the security requirement goals i.e. authentication, confidentiality, integrity, non-repudiation, freshness, availability and reliability. Therefore Table 8.1 Shows the comparison summary for IoT protocols with the present security protocol of this research based on security requirements

Table 8.1: Protocol Comparison based on Security Requirements

Security requirements addressed by the protocol	Authentication	Confidentiality	Integrity	Non-repudiation	Freshness	Availability	Reliability
[Figueroa et al., 2012]	×	√	√	×	√	√	**Good**
[Zhou et al., 2011]	√	√	×	×	×	×	**Limited**
[Khan et al., 2012]	√	√	×	×	×	√	**Good**
[Jara et al., 2011]	√	√	√	√	√	√	**Good**
[Kothmayr et al., 2012]	√	√	×	×	×	√	**Medium**
[Ikram et al., 2009]	√	√	√	√	×	√	**Medium**
[Brachmann et al., 2012]	×	×	√	×	√	×	**Variable**
[Raza et al., 2013]	×	×	√	×	√	×	**Medium**
[Kim H, 2008]	×	√	√	×	×	×	**Limited**
[Bonetto et al., 2012]	×	√	√	×	√	√	**Good**
[Raza et al., 2011]	×	√	√	×	√	√	**Medium**
[Jung et al. 2009]	√	√	√	×	√	√	**Good**
The Security Protocol for this research	√	√	√	√	√	√	**Good**

Furthermore, it has been provides a comprehensive comparison for the published work against the security protocol of this research, the comparison based on the security attacks, these attacks are eavesdropping, replayed, DoS, man in middle attacks, node capturing, selective forward, sinkhole, Sybil, wormhole and hello attack in Table 8.2 . These attacks can affect the security performance of IoT network.

Table 8.2 The Comparison Summary of 6LoWPAN Security Attacks for (outside and inside) Adversaries.

Security attacks addressed by the protocol	Protocols comparison based on security attacks									
	Eavesdrop	**Replayed**	**DoS**	**Man in Middle attack**	**Nod Capturing**	**Selective forward**	**Sinkhole**	**Sybil**	**Wormhole**	**Hello**
[Figueroa et al., 2012]	N/A	×	×	×	×	N/A	N/A	N/A	N/A	N/A
[Zhou et al., 2011]	√	√	√	√	√	×	×	×	×	√
[Khan et al., 2012]	√	√	√	√	√	×	√	√	√	N/A
[Jara et al., 2011]	√	√	√	√	√	N/A	√	√	√	√
[Kothmayr et al., 2012]	×	√	×	√	×	N/A	×	×	×	×
[Ikram et al., 2009]	×	√	√	√	√	N/A	√	√	√	√
[Brachmann et al., 2012]	×	×	×	×	×	N/A	×	×	×	×
[Raza et al., 2013]	×	×	×	×	×	N/A	×	×	N/A	N/A
[Kim H, 2008]	×	×	×	×	×	N/A	×	×	×	×
[Bonetto et al., 2012]	√	√	√	×	×	N/A	×	×	×	×
[Raza et al., 2011]	√	√	×	×	×	N/A	N/A	N/A	N/A	N/A
[Jung et al. 2009]	√	√	×	×	×	N/A	N/A	N/A	N/A	N/A
The Security Protocol for this research	√	√	√	√	√	√	√	√	√	√

Note: N/A means that there was not enough information to decide if the attack has been addressed by the protocol

In order to compare between different evaluation approaches, a broad set of evaluation approach has been used. Table 8.3 presents the evaluation of each IoT protocol. The following points explain each evaluation in details.

-Energy consumption: Energy is a critical resource of the network since it is one of the main elements that define the survivability of the network. Deciding if the protocol is suitable for a specific application highly depends on the energy consumption that occurs due to the protocol's functionality

-Communication overhead**:** Each protocol follows its own routing and security approach based on the application's objectives and requirements. These procedures create communication overhead with the messages that need to be exchanged between nodes during the setup establishment, data forwarding and maintenance phases of each procedure. Therefore, the protocols that evaluate how many messages are transmitted are considered under this criterion. This may concern security related packets, e.g. key setup packets.

-Storage overhead: The security approach taken by each protocol creates a storage overhead related to the size of cryptographic keys stored on each node. Aspects such as; power consumption, time, memory size etc play an important role in evaluating the protocols' design that assess the storage overhead of the new approaches.

Table 8.3 Evaluation of Protocol based on three fundamental aspects

Security Protocols	Energy consumption	Communication overhead	Storage overhead
[Figueroa et al., 2012]	√	×	√
[Zhou et al., 2011]	×	×	×
[Khan et al., 2012]	√	×	√
[Jara et al., 2011]	N/A	×	√
[Kothmayr et al., 2012]	×	×	√
[Ikram et al., 2009]	√	√	√
[Brachmann et al., 2012]	N/A	N/A	√
[Raza et al., 2013]	√	√	√
[Kim H, 2008]	N/A	N/A	×
[Bonetto et al., 2012]	√	×	√
[Raza et al., 2011]	√	√	×
[Jung et al. 2009]	√	×	√
The Security Protocol for this research	√	√	√

Note: N/A means that that is not enough information to decide if the energy consumption, communication overhead and storage overhead are addressed by the protocol

Chapter 9:

Conclusions and Further Work

9.1 Introduction

In this chapter, a concise summary of the main ideas proposed in this thesis is provided. The results, noteworthy achievements and the future applications of the proposed new concepts are similarly projected. It captures the main theme of this research study and shows how it succeeds in answering the research questions.

9.2 How are the key research questions addressed?

The research study has identified the crucial gaps in addressing the issue of providing security protocols in the IoT using LISP architecture. It also reveals the processes to integrate security protocols to the IoT in order to provide connectivity in LISP environments. Adding to this, and due to the open dynamic nature of the future of the IoT, for instance, the research highlights the need for addressing the issue of security of IoT based on LISP in different scenarios. These issues are then embedded into the four important research questions.

- **How to introduce an efficient architecture for the Internet of Things and what are the main operational entities that are required in this architecture?**

The answer to this question is to present a new LISP architecture in chapter 2. Furthermore, the new LISP architecture of the future internet has been introduced; the architecture defines the structure of the main networks entities, required for security purposes and for devices services. These entities are summarised as the following:

- Endpoint identifiers (EID) are represented in such devices as the IoT.
- Ingress Tunnel Router (ITR) is responsible for receiving packets from host and sending LISP packets towards the map server.
- Egress Tunnel Routers (ETR) are responsible for receiving LISP packets from map server and passing them to host

- X Tunnel Routers (XTR) act as both ETR and ITR.
- Mapping System (MS) is a globally database that contains all known ETR.

Consequently, The LISP has three important situations to allow the devices to join the network: the Initial Registration, Communications procedures and mobility procedures. These procedures have been discussed in details in section 2.10, chapter 2.

What are the Security vulnerabilities/threats in term of Internet of Things that the devices can be exposed to in Locator ID Separation Protocol?

As stated in chapter 3, to answer this question, the research has provided an investigation to the security issues that could occur from deploying the LISP in the IoT. Therefore, X.805 framework has been used to define the most security vulnerabilities and threats in the IoT.

- **How to provide End-to-End Security Commutation to Internet of Things?**

The answer to this question has been provided in chapter 6. The research has proposed a new security protocol communication to IoT devices that provides End-to-End Security. Where the achieved protocol has been designed via using El-Gamal encryption mechanism and is verified by AVISPA tool. The results demonstrate that they do not have any security flaws.

- **Considering the proposed security protocols, how could the security protocols interface to a refinement be integrated in order to approach a robust security for internet of things using Locator ID Separation Protocol network?**

The answer to this question is in Chapter 8. After designing the three security protocols in the Registration stage, resolving Stage and communication in chapters 5, 6, and 7, an interface for each level of these protocols has been set to achieve a robust security refinement protocol. The refinement protocol has been verified by using formal methods approach based on the AVISPA.

9.3 Main Contributions

As started in Chapter 1, the main contributions of this research study include:

- A critical review of existing network architectures for the IoT and network operation in these architectures has been presented. Furthermore, it reviews the existing solutions of different security protocols communication for the IoT. Consequently, this review uncovers important defects that hinder the successful realisation of designing End-to-End security communication for the IoT.
- Providing a comprehensive security analysis via using X.805 security framework to analyse the most security threats that have direct impact on the IoT, based on LISP network architecture. Besides, the study proposes a model concerning the security threats for the IoT based on LISP in three scenarios, i.e., initial Registration Stage, Resolving stage and communication procedure using AVISPA tool in order to design efficient protocols against these threats.
- Providing security enhancement protocols for the IoT based on LISP network architecture; these protocols are simulated via AVISPA verifying tool and are divided as the following:

  1. The Initial Registration Stage: each LISP protocol capable router needs to be registered with a map server known as Registration Stage.
  2. The Resolving addresses between the RLoC routers need to be addressed. For example, when the RloC router (A) wants to send data to the RLoC router (B), both of these routers need to be authenticated so that information can be reached from its original destination.
  3. Proposing a new End-to-End Security protocol for the IoT communication. The new security protocol provides a message authentication scheme that relies only on locally shared keys and symmetric cryptographic operation. It, also, introduces a level of security approximating End-to-End security mechanism. The foundation of the scheme's security is the creation of multiple disjoint (disconnected) authentication paths.

- Proposing an interface between each protocol, namely, the Registration stage, the Resolving stage and the communication procedure. The aim is to achieve security refinement protocol to the IoT based on LISP network architecture. Furthermore, a performance security refinement protocol analysis is provided in order evaluated the developed protocol impact on the IP-Based Sensor Network (IP-WSN).

- Provided a comparison of the recent security protocol for IoT against this present research study protocols.
- Verifying all the designed security protocols by using formal methods approach based on AVISPA tool. The performance and the evaluation have used Contiki and Cooja simulation tool to achieve the purpose

9.4 Elaboration on the main contributions

9.4.1 Identification of the main gaps in knowledge in the field of providing Internet of Things security in network environments.

The study conducts a comprehensive literature survey of related works in the IoT network and security. The study highlights the following crucial drawbacks in the investigated approaches:

- Uncertainty about the architecture of networks environments, which leads to many abstract solutions that do not reflect clear network architecture or/offer specific scenario solutions.
- Lack of routing efficacy: various research works have provided different types of architecture to support the IoT devices; however, they do not consider the huge numbers of the IoT devices that networks contain, which add, in turns, overloads on the routing table efficacy and network infrastructure.
- Poor realization of the IoT nature network, the unique features of the IoT should be considered in the designed security, therefore no conflict might result in implementing them.

9.4.2 Defining a security issues and a generic structure of Internet of Things Network architecture.

A generic architecture for the IoT network and security issues has been defined in Chapter 2 successively. The chapter discusse two important security issues that IoT devices suffer from #Issue 1, as a result of the vast increase vastly as well. This is because the communication between the devices can be easily exposed to disclosures by attackers. However, IPsec is not a practical solution to tiny devices link IoT. Hence, it is necessary to provide a mechanism in order to protect the transaction links between these devices against these adversaries. Concerning #Issue 2, IoT has a lack of privacy, integrity and confidentiality, defects that make the devices vulnerable against unauthorised device access. Likewise, chapter 2 introduce an architecture demonstrating the main operational network entities and transaction messages for each scenario. Therefore, the existing architectures network to IoT such as MIPv6, PMIPv6 and NEMO are discussed. In addition, certain some comparisons of the mobility protocols have been done in terms of different mobility and communication scenarios. The exchange messages used for neighbour discovery in IoT networks have also been investigated. The comparisons show that each mobility protocol has its own advantages depending on the scenarios it involves. Though, most of these mobility protocols, such as MIPv6, PMIPv6 and NEMO, where traffic is caused by these mobility protocols, may be too much for low-bandwidth wireless links in domains with large IoT. Besides, as the number of connected devices increase, i.e.IoT devices, the burden on the network infrastructure increase as well. One of the key challenges will be the size of the routeing tables and efficiency of the current routing protocols in the Internet backbone. For that reason, LISP supports different types of networks, i.e.IoT which defines compressed and size optimised mobility/communication signalling.

9.4.3 Defining a Security threats in Internet of Things

After defining the network architecture, security threats analyses has been provided in Chapter 3. The analysis expose the security threats in different levels on LISP architecture, i.e., initial Registration, resolving stage, communications procedures and mobility procedures. Specifically this researched study demonstrates the most common security issues and vulnerabilities in the IoT network. The analysis is based on the X.805 security standard and has, consequently, considered the security threats of IoT based on LISP network architecture. Furthermore, the analysis showes that there is a genuine need to provide new mechanisms to enforce Access control, Authentication, Non repudiation, Data Confidentiality, Communication Security, Data Integrity, Availability and Privacy. Additionally, the researched study has considered two layer threats namely, the infrastructure and service layers to expose the security threats during the unsecure signal transactions between the IoT, the LISP-capable routers and the mapping system. As results, a number of security vulnerabilities have been discovered and described.

9.4.4 Providing an Enhanced Security Protocol for Registration Stage in Locator/ID Separation Protocol Architecture

The enhanced security protocol for registration stage to the IoT based on LISP architecture has been already introduced in Chapter 4. The first designed version enhanced protocol used a Trust Authentication Server (TAS) as a third party trusted between ETR and MS, in order to provide two-party mutual authentication to entities i.e ETR and MS. Strikingly, attacks had been discovered by AVISPA; these attacks where divided into three categories in the registration protocol. (I) attacks on the I_ETR and I_TAS; here the intercepts the messages between ETR and TAS: These attacks are called playbacks. (II) Attacks on the I_MS and I_TAS, here the intruder listens to the conversation between MS and TAS. These attacks are called eavesdropping attack. (III) The third category is attack on I_ETR and I_MS where the intruder intercepts the communication and acts as I_ETR or I_MS conversation between the ETR and MS. These are known as Man in the Middle Attacks (MitM). Consequently, the final version of enhanced security protocol for registration stage in LISP architecture provided a new security method based on IBC, allowing a Map-Server to check the received information (i.e. the EID-Prefix) and provided secure authentication as well. The protocol is verified by AVISPA tool and the shows there are no security flaws.

9.4.5 Providing an Enhanced Security Protocol for Resolving Stage in Locator/ID Separation protocol Architecture

The enhanced security protocol for resolving stage to the IoT based in LISP architecture has been already introduced in Chapter 5. The first designed version of the enhanced protocol uses a random value generating and hash functions to provide a safe and secure transaction message between ITR and ETR. Furthermore, the protocol is simulated using AVISPA and two attacks have been discovered namely MitM and eavesdropping attacks. Thus, the final version of the enhanced security protocol for the resolving stage in LISP architecture has provided a new security method, based on Challenge-Response authentication and Key Agreement technique, allowing ITR and ETR to authenticate each other. The protocol is verified by AVISPA tool. The results show that this protocol is void of any security flaws.

9.4.6 Providing End-to-End Security Communication to Internet of Things using Locator/ID Separation protocol Architecture

Chapter 6 has provided End-to-End security communication protocol to IoT using LISP architecture. Furthermore, two security versions have been provided, the first version of security protocol used Challenge-Response and devices, when they communicate with each other based on LISP network architecture. A formal analysis was provided using AVISPA tool and MitM was discovered via AVISPA security tool. Besides, the final version of a new security protocol for IP-based Wireless Sensor (IP-WSN) communication has been proposed using El-Gamal encryption system. The security analysis and verification using AVISPA shows that no attacks have been found in the communication protocol.

9.4.7 Providing a Security Refinement Protocol and Performance to Internet of Thing using Locator/ID Separation protocol Architecture

Chapter 7 has provided a security refinement protocol and performance to the IoT devices using LISP architecture. Likewise, the refinement protocol is based on three security protocols namely resignation, resolving and communication which have been already introduced in previous chapters. Strikingly, the main idea of these protocols is to set an interface for each phase of the protocol to be joined together as one refinement protocol. Furthermore, the security refinement protocol analysis and verification using AVISPA show that no attacks have been detected. Finally, simulating the security refinement protocol in Contiki and Cooja simulation tool and the results show that the security refinement is highly scalable and the memory is quite efficient. It only needs 72 bytes of memory to storage the key in the device and it introduces 75.125 bytes of transmission and reception cost per connection. In this context, it has the advantage of securing defines against compromised nodes.

9.4.8 Providing a Comparison of Security Protocols to Internet of Things

Chapter 8 has provided a security protocols to IoT devices. The outcome of this analysis shows that the recent design for the security protocols has not met the security requirement goals i.e. authentication, confidentiality, integrity, non-repudiation, freshness, availability and reality. Besides, it has been provided a comprehensive comparison is for the published work against the security protocol of research, the comparison based on the security attacks, namely eavesdropping , relayed, Dos, MitM, node capturing, selective forward, sinkhole, Sybil, wormhole and hello attacks. These attacks can affect the security performance of IoT network. Also, it has been provided a comparison between different evaluation approaches such as energy consumption, communication overhead, and storage overhead.

9.5 Future improvements to solutions to maximize the study

The Internet of Things is a relatively new concept in terms of optimised protocols and security. It is an ever-changing area that will continue to change, and thus there is a lot of work for the future. Although, speed and cryptographic strength are especially important in the Internet of Things, however, the most pressing issue is simplifying the use of security in IoT for developers without a need for thorough knowledge of IT security. Designing and implementing security in protocols that are simple for developers to use is a must for the future of IoT. As devices in the Internet of Things are constrained devices, efficient implementations of cryptographic algorithms are especially important to keep the cryptographic strength at an acceptable level. Though the recommendations in this thesis are made with practical hypothesis for the future of IoT and encompass many different solutions, one will have to re-examine the recommendations with large changes in the market. As many cryptographic properties will always be existing and important, most of the recommendations will be the same for all foreseeable future. Consequently, the following is a list of set of improvements to the proposed solutions and related works upon which this study is based and which will hopefully result in a further improvement in performance:

- Defining the cryptographic algorithm in the proposed protocols: this involves comparing different algorithms and analysing how these might affect the performance of the whole protocol.
- Reducing the transaction messages of the proposed protocol in order to achieve lightweight protocol to the IoT.
- Implementing the proposed security refinement protocol in the IoT such as WSN device in order to evaluate the devices performerance.

9.6 Concluding Remarks

As shown throughout the different chapters of this thesis, the key issues of providing security refinement protocol to the IoT based on LISP architecture have been directly addressed. The security refinement protocol was verified using formal methods based on AVISPA tool and the archived results show that the refinement protocol is completely free of any security flaws. This significant achievement will definitely and hopefully will furnish the way for more development of the security protocols in the future of the Internet of networks. The outcome will definitely open new vistas to researchers to explore new possibilities in varied safe and secure use of the internet of network.

10. References

Aboelela E, (2007).,“Network simulation experiments manual: A systems approach, Morgan Kaufmann”.

Alejandro A, Hugues-Salas E, Haigh P, Marhuenda J, Price A, Sibson P, Kennard J, Erven C, Rarity J, Thompson M, Lord A, Nejabati R, Simeonidou D, 2016, ‘Secure NFV orchestration over an SDN –Controlled optical Network with Time-Shared Quantum Key Distribution Resources, Journal of Lightwave Technology, vol 35, no 8, pp 1357-1362.

Aris A, Oktug S, & Yalcin S, 2015, ‘Internet of Things security: Denial of Service Attacks’ IEEE Signal Processing and Communications Applications Conference, vol, no 152222639, pp 1-4.

Aouini L, Azzouz L, & Saidance L, 2016, ‘Using IPsec to secure multicast smart energy traffic’ International Conference on Performance Evaluation and Modelling in Wired and Wireless Networks, vol, no 16654468, pp 1-7.

Atzori L, Iera A, & Morabito G., 2010 ‘The Internet of Things: A Survey’ Computer Network, vol 54, no 15, pp 2787-2805

Atanasov L, Nikolov A, P& encheva E, 2017, ‘Study on generic functionality for quality of service control in M2M communications’ IEEE International Black sea Conference on Communications and Networks and Networking, vol, no 7901589, pp 1-5.

AVISPA- a tool for automated validation of internet security protocols, URL: http://www.avispa-project.org, [Date last accessed 23th May 2013]

Ayuso J, Marin L, Jara A, Skarmeta AFG, & Ayuso J. 2010 ‘Optimization of public key cryptography (RSA and ECC) for 8-bits devices based on 6LoWPAN’. 1st International Workshop on the Security of the Internet of Things (SecIoT’10), Tokyo, Japan.

Baker Z, Quinn H, 2016, ‘Design Test of Xilinx Embedded ECC for Micro Blaze Processors, IEEE Radiation Effects Data Workshop, vol, no 7891709, pp 1-7.

Bafandehkar M, Yasin S, Mahmod R, & Hanapi Z, 2013, ‘Comparison of ECC and RSA Algorithm in Resource Constrained Devices’, IEEE International Conference on IT Convergence and Security, vol, no 14047620, pp 1-3.

Beeharry J, Nowbutsing B, 2016, ‘Forecasting IPv4 exhaustion and IPv6 migration’ IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies, vol, no 16450362, pp 1-5.

Bernal-Hidalgo F, Garcia F, Lopez R, & Gomez-Skarmeta A, 2014, ‘A network access control solution based on PANA for intelligent transportation systems’ IEEE International Conference on Connected Vehicles and Expo’ vol, no 15522937, pp 444-449.

Biryukov A, Dunkelman O, Keller N, Khovratovich D, & Shamir 2010 ‘Key recovery attacks of practical complexity on AES-256 variants up to 10 rounds, in: Annu. Int.Conf. Theory Appl. Cryptogr.Tech., Springer, Berlin, Heidelberg, pp 299-319

Betzler A, Isern J, Garles C, Demirkol L, Paradells J, 2016, ‘Experimental evaluation of congestion control for CoAP communications without end-to-end reliability’, Ad Hoc Networks, vol, 52, no 1, pp 183-194

Bonetto R, Bui N, Lakkundi V, olivereau A, Serbanati A, Rossi M, 2012, ‘Secure communication for smart IoT objects: Protocol stacks, use cases and practical examples’ IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, vol, no 12933093, pp 1-7.

Boudguiga A, Olivereau A, & Oualha N, 2013, ‘Server Assisted Key Establishment for WSN: AMIKEY-Ticket Approach’ IEEE International Conference on Trust, Security and Privacy in Computing and Communications, vol, no 13971415, pp 94-101.

Brachmann M, Keoh S, Morchon O, & Kumar S, 2012, ‘End to End Transport Security in the IP-Based Internet of Things’ IEEE International Conference on Computer Communications and networks, vol, no 12965386, pp 1-5.

Burrows M, Abadi M, & Needham R (1990).,“A logic of authentication”, ACM Transactions on Computer Systems vol 8, pp 18–36. 1.

Butun I, 2017, ‘Privacy and trust relations in Internet of Things from the user point of view’ IEEE 7th Annual Computing and Communication Workshop and Conference, vol, no 16710270, pp 1-5.

Cao Y, & Bai J, 2015, ‘A passive attack against an asymmetric key Exchange Protocol’ IEEE International Conference on computer Science and Mechanical Automation, vol, no 15699028, pp 45-48.

Cheah W, & Liao C, 2017, ‘On Findability of Constrained Web of Things in a Smart Home Environment’ In International Conference on Platform Technology and Service vol, no 16773307,pp 1-6.

Chen, C & Wei, X (2016) ID Locator Split Based Solution for 5G Network URL: https://tools.ietf.org/html/draft-chen-id-locator-split-5g-solution-00 [Date last accessed 25th May 2016]

Cheng C, Lu R, Petzoldt A, Takagi T., 2017 ‘Secreting the Interns in a Quantum World’ IEEE Communications Society, vol 55, no 2, pp116-120

Christianah A, Boniface K, Aderonke F, Olufunso D, & Festus A, 2014,‘A robust multicast authentication scheme based on asymmetric key signature’ IEEE International Conference for Internet Technology and Secured Transactions, vol, no 14920218, pp 449-458.

Cisco-A, (2014) Locator/ID Separation Protocol LISP Cisco's The Network URL: https://newsroom.cisco.com/feature-content?articleId=1208342 [Date last accessed : 20th April, 2016].

Cisco-B, (2013) Connections Counter The Internet of Everything in Motion URL: https://newsroom.cisco.com/feature-content?articleId=1208342 [Date last accessed: 27th April 2016].

Contiki – The open source OS for the Internet of Things, URL: http://www.contiki-os.org/index.html [Date last accessed 20th May 2014]

DARPA, (2017) Defense Advanced Research Projects Agency. n.d. Retrieved, Available at: http://www.darpa.mil/about-us/budget [Date last accessed 14th April 2017].

Dawood H.A, & Jassim K.F, 2014, ‘Mitigating IPv6 Security Vulnerabilities’ International Conference on Advanced Computer Science applications and Technologies, vol,no 101109, pp 304-309.

David D, Jeyachandran A, 2016, ‘A comprehensive survey of security mechanisms in healthcare applications, IEEE International Conference on Communication and Electronics Systems, vol, no 16776447, pp 1-6.

Dooley M, Rooney T, 2013, ‘IPv6 Security Planning’ IPv6 Deployment and Management, vol1,no 9781118590447, pp 214.

Djeddai L, & Liu R, 2016, ‘IPSecOPEP: IPsec over PEPs architecture, for secure and optimized communications over satellite links’ IEEE International Conference on Software Engineering and Service Science, vol, no 16760510, pp 264-268.

Driessen B, Guneysu T, Kavun E, Mischke O, Paar C, & Popplmann T, 2012, ‘IPSecco: A lightweight and reconfigurable IPSec core’ IEEE International Conference on Reconfigurable Computing and FPGA, vol, no 13263891, pp 1-7.

Dragomir D, Gheorghe L, Costea S, & Radovici A, 2016, ‘A Survey on Secure Communication Protocol for IoT Systems’ IEEE International Workshop on Secure Internet of Things, vol, no 101109, pp 47-62.

Fanian A., Berenjkoub M., Saidi H, & Gulliver T,2010, ‘A scalable and efficient key establishment protocol for wireless sensor networks, in: IEEE Globecom Workshop on Web and Pervasive Security. Vol, no 11774577, pp 1533- 1538

Fei, N, Xuefen,C, Wen, D, & Haifeng, Y 2016, ‘Jitter analysis of real-time services in IEEE 802.15.4 WSN and Wired IP concatenated networks’, The Journal of china Universities of Posts and Telecommunications, vol. 23,no. 4.pp.11-8

Ferguson N, Schneier B, & Kohno T, 2010 ‘Cryptography Engineering: Design principles and practical Applications’ John Wiley & Sons ISBN: 978-0-470-4742-2

Figueroa P, Perez J, Amezcua I, & Hernandez V, 2012, ‘A Lightweight and secure protocol to Access web services in 6LoWPAN’ IEEE the Electrical Communications and Computers (CONIELECOMP) International Conference, vol, no 978-1-4577-1326-2, pp 80-85,

Fioreze T, & Heijenk G, 2011, ‘Extending the Domain System (DNS) to provide geographical addressing towards vehicular ad-hoc networks’ IEEE vehicular Networking Conference, vol, no 12461127 pp 70-77.

Fuller V., & Farinacci D., (2013) “Locator/ID Separation Protocol (LISP) Map-Server Interface”, Internet Engineering Task Force (IETF), RFC 6833, URL: http://www.ietf.org/id/draft-ietf-lisp-ms-16.txt , [Date last accessed on 10 Feb 24 Mar 2013]

Gaithuru J, Bakhtiari M, Salleh M, & Muteb A, 2015, ‘A comprehensive literature review of asymmetric key cryptography algorithms for establishment of the existing gap’ IEEE Malaysian Software Engineering Confernce, vol, no 16005524, pp 236-244.

Gartner, J (2013) the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020 URL: http://www.gartner.com/newsroom/id/2636073 [Date last accessed 1th May 2016]

Giridhar R, & Suresh S, 2016, ‘Cluster based Certificate Authority Scheme in WSN’ IEEE International Conference on Wireless Communications, Signal Processing and Networking, vol, no 16303908, pp 367-371.

Haripriya A, Kulothungan K, 2016, ‘ECC based self-certified key management scheme for mutual authentication in Internet of Things, IEEE International Confernce on Emerging Technological Trends, vol, no 16726928, pp 1-6.

Hasan K, Sharif T, Hossain S, & Atiquzzaman M, 2017,‘Comparison of Nemo Schemes in proxy Mobile IPv6 Domain’ IEEE Global Communications Conference, vol, no 1665329, pp 1-6.

Hernndez-Ramos J Carrillo D, Lopez R, Skarmeta A, 2015, ‘Dynamic security credentials PANA-based provisioning for IoT smart objects, IEEE 2nd World Forum on Internet of Things, vol, no 15729146, pp 1-6.

Hossain E., 2009 “Introduction to network simulator ns2”, Springer,.

Huang C, Chiang M, Dao D, & Pham B, 2017, ‘A group-based fast media independent handover control scheme for proxy mobile IPv6 (GB-FMIH-PMIPv6)’ IEEE International Black Sea Conference on Communications and Networking, vol, no 7901592, pp 1-5

Huany Q, & Li H, 2017, ‘An efficient Public –key searchable encryption scheme secure against inside keyword guessing attacks’, Information Sciences, vol 403-404, no, pp 1-14

Hummen R., Wirtz H., Ziegeldorf J., Hiller J., & Wehrle J, 2013,“Tailoring end-to-end IP security protocols to the Internet of things”, IEEE International Conference on Network Protocols, vol, no 14082474, pp 1-10.

Husamuddin M, & Qayyum M, 2017, ‘Internet of Things: A study on security and privacy threats’ IEEE International Conference on Anti-Cyber Crimes, vol, no 7905270, pp 1-5.

Hussen R., Tizazu A, Miao T., Takkyeun L., Youngjun C., and Ki-Hyung K, 2013, ‘SAKES: secure authentication and key establishment scheme for M2M communication in the IP-based wireless sensor network (6LoWPAN)’,IEEE International Conference on Ubiquitous and Future Networks (ICUFN), vol, no 13840506, pp 246-251.

Imran S, Ko & Y, 2016 ‘Optimizing fast handover in MIPv6 through buffered packet forwarding and out of sequence packets reduction’, International Conference on Information and Communication Technology Convergence, vol,no 1651997, pp 156-161

Ikram M, Chowdhury A, Zafar B, SooCha H, Kim K, whaYoo S, & KyooKim S, 2009, ‘A simple lightweight authentic bootstrapping protocol for IPv6-based low rate wireless personal area networks (6LoWPANs), International conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly; vol, no, pp 937-941

Isah M, & Ewards C, 2015, ‘Inter-domain with LISP-MN –A performance Comparison with MIPv6’ IFIP Wireless and Mobile Networking Conference, vol, no 15758169, pp 80-87

Jara A, Marin L, Skarmeta G, Singh D, Bakul G, Daeyeoul K, 2011,‘Secure Mobility Management Scheme for 6LoWPAN ID/Locator Split Architecture’ IEEE International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, vol, no 12172054, pp 310-315

Jayapandia N, Rahman A, Radhikadevi S,& Koushikaa M, 2016, ‘Enhanced cloud security framework to confirm data security on asymmetric and symmetric key encryption’ IEEE world conference on Futuristic Trends in Research and Innovation for Social Welfare, vol, no 16358577, pp 1-4.

Jicha R, Patton M, & Chen H, 2016, ‘Identifying devices across the IPv4 address space’ IEEE conference on Intelligence and Security informatics, vol, no 16467704, pp 199-201.

Jung W, Hong S, Ha M, Kim Y, Kim D, 2009, 'SSL-Based Lightweight Security of IP-Based Wireless Sensor Networks' IEEE International Conference on Advanced Information Networking and Applications Workshops, vol, no 10733765, pp 1112-1117.

Karras D, 2012, 'On defining an hierarchical Secure proxy agent architecture for embedded communication network applications' Telecommunications forum, vol, no 13264908, pp 131-134.

Kim, H 2008, "Protection against Packet Fragmentation Attacks at 6LoWPAN Adaptation Layer'' Convergence and Hybrid Information Technology,. ICHIT 08.Interational Conference, vol.1, no., pp.796-801.

Khan R, & Mir A, 2014, 'Sensor fast proxy mobile IPv6 (SFPMIPv6)-A framework for mobility supported IP-WSN for improving QoS and building IoT, International Conference on Communication and Signal Processing, vol, no 14754864, pp 1593-1598.

Khan R, & Shiranzaei A, 2017, 'IPv6 Security tools-A Systematic review' International Conference on Computing Communication and Automation vol; no 16585524, pp 459-464.

Khan S., Pastrone C., Lavagno L., Sporito M., 2012,'An Authentication and Key Establishment Scheme for the IP-Based Wireless Sensor Network'' The 7th International symposium on Intelligent Systems Techniques for Ad hoc and Wireless sensor Network (IST-AWSN), vol 10, no, pp 1039-1045

Khan W, Zangoti H, Aalsalem M, Zahid M, & Arshad Q, 2016,'Mobile RFID in Internet of Things: Security attacks, privacy risks, and countermeasures' IEEE International Conference on Radar, Antenna Microwave, Electronics, and Telecommunication's, vol, no 16670039, pp 36-41.

Kim H, 2008, 'Protection Against Packet Fragmentation Attacks at 6LoWPAN Adaptation Layer' IEEE International Conference on Convergence and Hybrid Information Technology, vol, no 10187301, pp 796-801.

Klodowski K, Lukasik P, Kryzak A, 2016, 'Approximation of the actual spatial distribution of the b-matrix in diffusion tensor imaging with bivariate polynomials' IEEE Federated Conference on computer Science and Information Systems, vol, no, 16428554, pp 943-946.

Kushalnagar N, Montenegro G, Schumacher C. (2007) RFC 4919: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals URL: http://www.tools.ietf.org/html/rfc4919 [Date last accessed 2th May 2017]

Kundu A, 2015 'Mitigation of Storage Convert Channels in IPSec for Qous Aware Applications' vol 54 ,pp 102-107

Kogatam S.,(2014) "Internet of Things Enabling Technologies RFID". URL: http://www.slideshare.net/swethak36/iot-rfid-finalppt [Date last accessed 23 May 2016]

Kortuem G, Kawsar F, Fitton D, & Sundramoorthy V., (2010) " Smart objects as building blocks for the Internet of Things' IEEE Internet Comping, vol 14, no 1, pp 44-51.

Kothmayr T, Schmitt C, Hu W, Brunig M, & Carle G, 2012, 'A DTLS based end-to-end security architecture for the Internet of Things with two-way authentication' IEEE Conference on Local Computer Networks, vol, no 13285725, pp 956-963.

Kothmay T, Schmitt C, Hu W, Bruning M, & Carle G, 2013, 'DTLS based security and two-authentication for the Internet of Things' Ad Hoc Networks, vol 11, no 8, pp 2710-2723.

Lara-Nino C, Diaz-Perez A, & Morales-Sandoval-M, 2017, 'Lightweight Hardware Architectures for the Present Cipher in FPGA' IEEE Transactions on Circuits and Systems I : Regular Papers, vol 99, no 2686783, pp 1-12.

Levi A, & Sarimurat S, 2017, 'Utilizing hash graphs for key distribution for mobile and replaceable interconnected sensors in the IoT context' Ad Hoc Network, vol 57, no, pp 3-18.

Li F, Han, Y & Jin,C 2016 'practical access control for sensor network in the context of the internet of things' Computer Communications, vol.89-90, pp.154-164

Liu B, & Wu H, 2016, 'Efficient multiplication architecture over truncated polynomial ring for NTRUEncrypt system' IEEE International Symposium on Circuits and Systems , vol, no 16214439, pp 1174-1177.

Li X, Ibrahim M, Kuman S, Sangaiah A, Gupta V,& Choo K., 2017 'Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area network' Computer Networks vol 26, no 2, pp 181-201.

Liu Z, Liu H, Hardy J, & Liu L, 2016, ‘Uniform Resource Identifier Encoded by Base62x’ IEEE International Conference on ubiquitous Intelligence and Computing, vol no 16158388, pp 1087-1093.

Maino, F., Ermagan, V., Cabellos, A., Saucez, A., & Bonaventure, O.: (2012) ‘LISP-Security (LISP-SEC)’ URL: https://tools.ietf.org/html/draft-ietf-lisp-sec-12 Internet-Draft , [Date last accessed on 10 Feb 2013].

Mahalle P.N, Prasad N.R, & Prasad.R., 2014, ‘Novel Threshold Cryptography-based Group Authentication (TCGA) Scheme for the Internet of Things (IoT)’ International conference on wireless communications, Vehicular Technology, Information Theory and Aerospace & Electronics Systems (VITAE), Vol, no 14697044, pp 1-5.

Makinen S.J, 2014, ‘Internet of Things disrupting business ecosystems: A cause in home automation’ IEEE International Conference on Industrial Engineering and Engineering Management, vol, no 1483755 ,pp 1-5.

Masdari M, & Ahmadzadeh S, 2017, ‘A survey and taxonomy of the authentication schemes in Telecare Medicine Information Systems’ Journal of Network and computer Applications, vol 87, no 1, pp 1-19.

Meca F., Ziegeldorf J, Sanchez P., Morchon O., Kumar S., & Keoh S., 2013, “HIP security architecture for the IP-based Internet of things’, in: 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), vol, no 13615018, pp 1331-1336.

Mehmood A, Umar M, & Song H, 2017, ‘ICMDS: Secure-cluster multiple-key distribution scheme for wireless sensor networks’ Ad Hoc Networks, vol 55, no, pp 97-106

Mikhail M, Abouelseoud Y, & Eikobrosy G, 2014 ‘Extension and application of El-Gamal encryption scheme’ Computer Applications and information Systems, vol 10, no 978-4799-3351-8

Miyaji A, & Omote K, 2013, ‘How to Build Random Key pre-distribution Schemes with self-Healing for Multiphase WSNs’ IEEE 27th International Conference on Advanced Information Networking and Applications, vol no 13581034, pp 205-212.

Mortazavi S, Pour A, & Kato, T, 2011, ‘An efficient distributed group key management using hierarchical approach with Diffie-Hellman and Symmetric algorithm: DHSA’ IEEE International Symposium on Computer Networks and Distributed Systems, vol, no 11976038, pp 49-54

Mstafa R, Elleithy K, & Abdelfattah E, 2017 ‘A Robust and Secure Video Steganography Method in DWT- DCT Domains Based on Multiple object Tracking and ECC’ vol 99, pp 1-1

Mulligan G., William C., and Huo D., 2010, "Mobility Considerations for 6LoWPAN" - draft- williams-6lowpan-mob-02.txt, March 8, 2010 URL at: https://tools.ietf.org/html/draft-williams-6lowpan-mob-02 [Date last accessed 10 Feb 2016].

Nawir M, Amir A, Yaakob N, & Lynn O,2016, ‘Internet of Things (IoT): Taxonomy of Security attacks’ IEEE International Conference on Electronic Design, vol, no 7804660, pp 321-326

Ngoepe P, Meyer W, Auret W, & Diale E, 2017, ‘DLTS characterization of defects in GaN induced by electron beam exposure’, Materials Science in Semiconductor Processing, vol 64, no 1, pp 29-31.

Nicolls V, Le-Khac N, Chen L, Scanlon M, 2017 ‘IPv6 Security and forensics’ International Conference on Innovative Computing Technology, vol, no 16656141, pp 743-748.

Nikolenko S, Kogan K, Retvari G, Berczi-Kovacs E, & Shalimov A, 2016,‘ How to represent IPv6 forwarding tables on IPv4 or MPLS data places’ IEEE Conference on Computer Communications Workshops, vol, no 16285661, pp 1-6.

Nimala S,& Praveena R, 2016, ‘Implementation of efficient modular bypass multiplier logic in RSA cryptographic Processor’ IEEE International Conference & Workshop on Electronics & Telecommunication Engineering, vol, no 16584784, pp 52-57.

Nurse J, Erola A, Agrafioties L, Goldsmith M, & Cressse S, 2015, ‘Smart Insiders: Exploring the Threat from Insiders Using the Internet of Things’ IEEE International Workshop on Secure Internet of Things, vol, no 15791224, pp 5-14.

Papadimitratos P, & Deng J, 2012, 'Stealthy pre-attacks against random key pre-distribution security', IEEE International Conference on Communications, vol, no 13153967, pp 955-959

Pawlowski M, Jara A, Ogorzalek M, 2015, 'Maximizing the Extensible Authentication Protocol Mazimum Transfer Unit to Minimze the Authenticating Data Transmission in the IEEE 802.15.4 Networks' IEEE Global Communications Conference, vol, no 15821042, pp 1-6.

Paris C, Kelbe D, Aardt J, & Bruzzone L, 2017, 'A novel Automatic Method for the fusion of ALS and TLS LiDAR for Robust Assessment of tree crown structure, IEEE Transactions on Geoscience and Remote sensing, vol, 99, no 2675963, pp 1-15.

Pishva D, 2017, 'Internet of Things: Security and privacy issues and possible solution' vol, no 16777286, pp 1-12.

Pradeska N, Widyawan W, Najib W & Kusumawardani S, 2017, 'Performance analysis of objective function MRHOF and OF0 in routing protocol RPL IPv6 over low power wireless personal area networks (6LoWPA) vol, no 16707930, pp 1-6.

Prakasha K, Muniyal B, Kavyashree K, & Deeksha D, 2016, 'Electrocardiogram-Kerberos authentication scheme for secure services' IEEE International Conference on Inventive Computation Technologies, vol, no 16620664, pp 1-4.

Qin B, Liu, Sun S, Deng R, & Gu D, 2017, 'Related-key secure key encapsulation from extended computational bilinear Diffie-Hellman' Information Sciences, vol 406-407, no 1, pp 1-11.

Raheem A, Lasebae A, & Loo J, 2014, 'A secure Authentication for IP-Based Wireless Sensor Communications using the Location/ID split Protocol (LISP)' International conference on Trust Security and Privacy in Computing and Communications, vol, no 14856103, pp 840-845.

Raheem A, Lasebae A, Aiash M, & Loo J, 2013, 'Supporting communication in the IoT using the Location/ID split protocol: A security analysis' Second International Conference on Future Generation Communication Technologies, vol, no 14162143, pp 143-147

Rahman H, Islam, N, Jany M, Shariful S, & Rahman M, 2017, ‘Multimedia content security with random key generation approach in cloud computing’ IEEE International Conference on Imaging vision & Patten Recognition, vol, no 16777706, pp 1-6.

Rantos K, Papanikolaou A, Manifavas C, & Papaefstathiou L, 2014, ‘IPv6 Security for Low power and lossy Network’ IFIP wireless Days IEEE, vol, no 14000985,pp 1-8.

Rao M, Coleman J, Newe T, 2016, ‘An FPGA based reconfigurable IPSec ESP core suitable for IoT applications’ Internet Conference on Sensing Technology, vol, no 16575324, pp 1-5.

Raza S, Duquennoy S, Chung T, Yazar D, Voigt T, Roedig U, 2011, ‘Securing communication in 6LoWPAN with compressed’ IEEE International Conference on Distributed Computing in Sensor Systems and Workshops, vol, no 12180908, pp 1-8.

Raza S, Duquennoy S, Voigt T, & Roeding U, 2011, “Demo abstract :Securing communication in 6LoWPAN with compressed IPsec”, in: International Conference on Distributed Computing in Sensor Systems and Workshop, vol,no12206087, pp 1-2.

Raza S, Shafagh H, Hewage K, hummen R, & Voigt T, 2013, ‘Lithe: Lightweight Secure CoAP for the Internet of Things’ IEEE Sensors Journal, vol 13, no 13747698, pp 3711-3720

Raza S, Thorat P, Challa R, & Choo H, 2017. ‘on demand inter domain mobility in SDN based Proxy Mobile IPv6’ International Conference on Information Networking, vol, no 7899503, pp 194-199.

Reegan S, & BaburaJ E, 2016, ‘Polynomial and multivariate mapping-based triple-key approach for secure key distribution in wireless sensor networks, computers & Electrical Engineering, vol,no,pp 1-17.

Roman R, Alcaraz C, Lopez J, & Sklavos N, 2011, ‘Key management system for sensor networks in the context of the internet of Things’ Computer & Electrical Engineering, vol 37, no 2, pp 147-159.

Sahraui S, & Bilami A, 2014, ‘Compressed and distributed host identity protocol for end-to-end security in the IoT’ IEEE International Conference on Next Generation Networks and Services, vol, no 14820525, pp 295-301.

Saikia M, & Hussain A, 2016, 'Improving the performance of Key pre-distribution scheme in sensor network using clustering of combinatorics' IEEE International Conference on Computing Communication and Automation, vol, no 16598900, pp 682-686.

Sanchez J., 2015 'Analysis of the Security IPv6 and Comparative study between Two Routing Protocols Oriented to IPv6' Asia-Pacific Conference on computer Aided System Engineering, vol,no15490593, pp374-379.

Shaharuddin U, Rahman R, Kassim M, & Yusof M, 2017, 'Performance comparison of Multimedia Applications over IPv4 and IPv6 Dual Stack technology' International Conference on System Engineering and Technology, vol, no 16670090,pp 1-6

Sharma V, & Hussasin, 2016, 'Node authentication in WSN using key mechanism' IEEE International Conference on ICT in Business Industry & Government, vol, no 7892691, pp 1-7.

Shelby Z., Bormann. C, (2009) "6LoWPAN: The Wireless Embedded Internet", ISBN9780470747995 (H/B)

Shuai Y, & Qianli Xing L, 2016, 'A tunnel broker based IPv6 access system from a small scale network with IPv4 upstream, IEEE Information Technology, Networking, Electronic and Automation Control Conference, vol, no 16284299, pp 206-210.

Siddika F, Hossen M, & Saha S, 2017, 'Transition from IPv4 to IPv6 in Bangladesh: The competent and enhanced way to follow' International Conference on Networking, Systems and security, vol, no 16774665, pp 1-6.

Singh B, Singh, T, & Sachdeva H, 2017, 'Evaluating the Performance of density grid-based clustering using ABC technique for efficient routing in WSNs' IEEE Annual Conference on Information Sciences and Systems, vol, no 7926099, pp 1-6

Stallings W., (2011), "Cryptography and Network Security: Principles and Practice", 5th Edition., ISBN 13:9780136097044.

Soliman S, Magdy B, & Abd EL Ghany M, 2016, 'Efficient implementation of the AES algorithm for security applications' IEEE International System on Chip Conference, vol, no 7905466, pp 206- 2010.

Sundmaeker H., Guillemin P., Friess P., and Woelffle S., 2010., 'Vision and Challenges for Realising the Internet of Things'. European cluster CERP-IoT, European Union, ISBN: 978-92-97-15088-3.

Takahashi A, Maeda T, & Okabe Y, 2012, 'Design and Implementation of a Secure Public Wireless Internet Service Model Using Host Identity Protocol' IEEE/IPSJ 12th International Symposium on Applications and the Internet, vol, no 13000515, pp 19-28.

Tan C, Prabowo T, & Le D, 2016 'Breaking an ID-Based encryption based on discrete logarithm and factorization problems' Information Processing Letters, vol 116, no 2, pp 115-119

Tbatou Z, Asimi A, Asimo Y, & Sadqi Y, 2015, 'Kerberos V5: Vulnerabilities and perspectives, IEEE Third World Conference on Complex Systems, vol, no 16052588, pp 1-5.

Tong W, Liang J, jin X, & Li Z, 2013, 'A survey on key pre-distribution scheme of distributed WSNs' IEEE Third International Conference on Instrumentation, Measurement, computer, communication and control, vol, no 101109, pp 242-246.

Twayej W, & Al-Raweshidy H, 2017, 'M2M energy efficiency routing protocol MLCMS by using 6LoWPAN based on IoE' IEEE International Black sea Conference on Communications and Networking, vol, no 7390151, pp 1-5.

Uahhabi Z., & Bakkali H, 2016, 'An approach for evaluating trust in X.509 Certificates', IEEE International Conference for Internet Technology and Secured Transactions, vol, no 166674920, pp 196-203.

Walsh K, 2016, 'TLS with trustworthy certificate authorities' IEEE Conference on Communications and Network Security, vol, no 16692755, pp 1-9.

Wang D, Cheng H, He D, & Wang P, 2016, 'on the challenge in designing identity-based privacy preserving authentication schemes for mobile devices, IEEE systems journal, vol 99 no 2585681, pp 1-10.

Xiaorong F, Jun L, Shizhun J, 2013, 'The research on mobile Ipv6 Security features' IEEE Symposium on Wireless Technology and Applications, vol, no 13974439, pp 125-128.

Yajam H, Ahmadabadi Y,& Akhaee M, 2016, 'Deniable Encryption based on Standard RSA with OAEP' IEEE International Symposium on Telecommunications, vol, no 1675858, pp 84-88.

Yang L, Ding C, & Wu M, 2013, 'Establishing Authenticated Pairwise Key using Pairing-based Cryptography for Sensor Networks', IEEE International Conference on Communications and Networking in China, vol, no 14022510, pp 517-522.

Yang Y, Peng H, Li L,& Niu N, 2016, 'General Theory of Security and a Study Case in Internet of Things', IEEE Internet of Things Journal, vol, no 2597150, pp 592-600.

Yang Y, Wu L, Yin G, Li L, & Zhao H, 2017, 'A Survey on Security and Privacy Issues in Internet of Things' IEEE Internet of Things Journal, vol PP, no 99, pp 1-1.

Yang Z, Xu W, & Xu H., 2016 'Energy Efficient Non-orthogonal Multiple Access for Machine-to-Machine Commutations' IEEE Communications Letters, vol 21, No 4, pp 817-820.

Yao X, Chen Z, & Tian Y, 2015, 'A lightweight attribute-based encryption scheme for the Internet of Things' Future Generation computer Systems, vol 49, no, pp 104-112.

Yeh J, Su P, Huany S, Chiang T, 2016, 'SNAKE game Al: Movement rating functions and evolutionary algorithm-based optimization' IEEE conference on Technologies and Applications Artificial Intelligence, vol, no 16758298, pp 256-261.

Ye M, Zhang X, Racz G, Jiang Q, & Moret B, 2017, 'NEMo: An Evolutionary Model with Modularity for PPI Networks' IEEE Transactions on NanoBioscience, vol, no 2656058, pp 1-1.

Yin A, Chen D, Ding Y, 2014, 'An efficient and secure authentication scheme based on NTRU for 10G Ethernet passive optical', Optik- International Journal for Light and Electron Optics, vol 125, no 24,pp 7207-7210.

Yu Z, Wang Q, Zhang W, & Dai H, 2015, 'A could certificate Authority Architecture for Virtual Machines with Trusted Platform Module' IEEE International Conference on High Performance Computing and Communications, vol no 268, pp 1377-1380.

Zaidan B.B, Zaidan A.A, Al-Frajat A.K & Jalab H.A., 2010 'on the Differences between hiding Information and Cryptography Techniques: An overview' vol 10, no 15,pp 1650-1655

Zhang N, Wu X, Yang C, Shen Y, Cheng Y, 2016, 'A lightweight authentication and authorization solution based on Kerberos' IEEE Advanced information Management Communications, Electronic and Automation Control Conference, vol, no 16709279, pp 742-746.

Zhang Y, Clouet A, Awotayo O, Davids C, & Gurbani V, 2015, 'Benchmarking the session imitation protocol (SIP)' IEEE International Workshop Tehnical Committee on Communications Quality and Reliability, vol, no 15240970, pp 1-6.

Zhou Y,Ja Z., Sun X. ,Li X & Ju L, 2011, ''Design of Embedded Secure Gateway Based on 6LoWPAN'' IEEE International Conference on communication technology, vol, no 12575725, pp 732-736.

Zigler S, Skarmeta A, Kirstein P, & Ladid L, 2015.,"Evaluation and recommendations on IPv6 for the Internet of Things", in: Internet of Things (WF-IoT), IEEE 2nd World forum on, IEEE ,pp.548-552.

APPENDIX –A

Registration Protocol version-I

In HLPSL Code

```
% Role of the initiator by ETR:
role router_ETR (ETR, MS: agent,
           PUK_ETR: public_key,
           PSK1_TAS, PSK2_TAS: Symmetric_key,
           H: hash_func
           KeyRing: (agent.public_key) set,
           Snd, Rcv: channel(dy))

played_by ETR def=

  local State : nat,
       N1, N2: text,
       Map-Register: text,
       Map-Notify: text,
       ACK: text,
       PUK_MS: public_key

  init State := 0

  transition

   % Start, if ETR must request MS public key from TAS
   ask.    State  = 0 /\ Rcv(start) /\ not(in(MS.PUK_MS', KeyRing))
      =|> State':= 1 /\ Snd {(ETR.N1.Map-Register),{H(N1.Map-  Register.MS)}

   % Receipt of response from Trust Authentication Server
   learn.  State  = 1 /\ Rcv {TAS,N1,Map-Register), ({PUK.MS'}_inv(PSK1_TAS))
      =|> State':= 0 /\ KeyRing':=cons(MS.PUK', KeyRing)

   % Start/resume, provided ETR knows MS public key
   knows.  State  = 0 /\ Rcv(start) /\ in(MS.PUK', KeyRing)
      =|> State':= 4 /\ ETR':=new() /\ Snd {(ETR,Map-Register)   {MS}_PUK')
                  /\ secret(ETR',setra,{ETR,MS})
                  /\ witness(ETR,MS,Router,Map_Server)

   cont.   State  = 4 /\ Rcv{(MS.Map-Notify)_PUK_ETR)}
      =|> State':= 6 /\ Snd {(ETR.ACK.Map-Notify)_PUK_MS)}


end role

% Role of the receiver by MS:
role Map_Server(ETR, MS: agent,
        PUK_MS: public_key,
        PSK1_TAS, PSK2_TAS: Symmetric_key
        H: hash_func
        KeyRing: (agent.public_key) set,
        Snd, Rcv: channel(dy))
played_by MS def=

  local State: nat,
       N1, N2: text,
       PUK_ETR: public_key
       Map-Register: text,
       Map-Notify: text,
       ACK: text,

  init State := 2
```

```
  transition

   % Start if MS must request ETR public key from key server
   ask.    State  = 2 /\ Rcv {(ETR,Map-Register)(PUK_MS)} /\ not(in(ETR.PUK_ETR', KeyRing))
       =|> State':= 3 /\ Snd{(MS.N2.Map-Register),{H(N2.Map-  Register.ETR)}

% Receipt of response from Trust Authentication Server
   learn.  State  = 3 /\ Rcv {TAS,N2,Map-Register), ({PUK.ETR'}_inv(PSK2_TAS))       =|> State':= 2 /\
KeyRing':=cons(ETR.PUK', KeyRing)

   % Start/resume, provided MS knows ETR public key
   knows.  State  = 2 /\ Rcv /\ in(ETR.PUK', KeyRing)
       =|> State':= 5 /\ MS':=new() /\ Snd{(ETR,Map-Notify)   {ETR}_PUK')                    /\
secret(MS',smsa,{MS,ETR})
                    /\ witness(MS,ETR,Map_Server,Router)

   cont.   State  = 5 /\ Rcv {(ETR.ACK.Map-Notify)_PUK_MS)})

end role

% Role of the Trust Authentication Server
role server(TAS: agent,
           PSK1_TAS, PSK2_TAS: Symmetric,
           PUK_ETR, PUK_MS: Public key
           H: hash_func
           KeyMap: (agent.public_key) set,
           Snd, Rcv: channel(dy))
played_by S def=

  local ETR, MS: agent,
        State: nat,
        PUK_MS: public_key

  init State := 8

  transition
   req1.   State = 8 /\ Rcv{(ETR,N1,Map-RequestRegister),{h(N1,Map-Register)'.MS') /\ in(MS'.PUK_MS',
KeyMap)
       =|> State':= 8 /\ Snd({N1,Map-Register)'.PUK_MS'}_inv(PSK1_TAS))

Req2.   State = 9 /\ Rcv{(MS,N2,Map-Register),{h(N2,Map-Register)'.MS') /\ in(ETR'.PUK_ETR', KeyMap)
       =|> State':= 9 /\ Snd({N2,Map-Register)'.PUK_ETR'}_inv(PSK2_TAS))

end role

% Role representing a partial session between ETR and MS:
role nspk(Snd, Rcv: channel(dy),
         PSK1_TAS, PSK2_TAS: Symmetric
         Instances: (agent.agent.Symmetric_key.Symetric_key) set,
         KeySet: agent -> (agent.Symmetric_key) set)
def=

  local ETR, MS: agent,
        PUK_ETR, PUK, MS: public_key,

 composition
    /\_{in(ETR.MS.PUK_ETR.PUK_MS,Instances)}
      (ETR(ETR,MS,PUK_ETR,PSK1_TAS,KeySet(ETR),Snd,Rcv)
      /\ MS(MS,ETR,PUK_MS,PSK2_TAS,KeySet(MS),Snd,Rcv))
```

```
end role



% The main role:
role environment() def=

  local KeyMap: (agent.public_key) set,
        Snd, Rcv: channel(dy)

  const etr, ms, tas, i: agent,
        PUK_ETR, PUK_MS, KUPi: public_key,
        PSKi, PSK1_TAS, PSK2_TAS: Symmetric,

        snetr, snms, router_map_server_nms, map_server _router_netr: protocol_id

  init KeyMap := {etr.puk_etr, ms.puk_ms, i.kupi}

  intruder_knowledge = {etr, ms, psk1_tas, psk2_tas, puk_etr, puk_ms, puki,pski, inv(ki)}

  composition
        map_server(ms,psk1_tas, psk1_tas KeyMap, Snd, Rcv)
     /\ nspk(Snd, Rcv,                 % channels
            Psk1, psk2                 % Pre-shared keys to secure
                                       % the connections between
                                       % the TAS and ETR, MS
            {etr.ms.puk_etr.puk_ms,    % session instances
             etr.i.puk_etr.ki,
             i.ms.ki.puk_ms
            },
            {etr.{etr.puk_etr,ms.puk_ms},   % initial
             KeyRings
             ms.{ms.puk_ms},
             i.{i.ki}})
end role

% Description of goal properties:
goal

  secrecy_of sn1, sn2,etr, ms, tas
  Weak_ authentication on etr_tas
  Weak_authentication on ms_tas
  Weak_authentication on etr_ms
  authentication_on router_map_server_n1
  authentication_on map_server_router_n2

end goal
```

APPENDIX –B

Resolving Procedure Protocol Version-I

In HLPSL Code

```
role router_ITR (ITR, ETR, MS : agent,
                    PSK_ItrMs,PSK_MsEtr: symmetric_key,
                                 H    : hash_func,
                             Snd, Rcv  : channel(dy),

played_by ITR
def=

local  State         : nat,
SK : symmetric_key
                 N1,N2 : text
   MappRequest,MappReply: text
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id

init   State := 0

transition
   1. ask.State = 0 /\ Rcv(start) =|>
State' := 1
       /\  Snd(ITR.MS.{ITR.N1.MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_ItrMs)
   2. learn.State = 1
       /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
       /\ witness(ITR,ETR,MS,N1,N2')
       /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})
3.knows.State = 2
       /\ Rcv(ETR') =|>
State' := 3 /\ Snd(ETR.ITR.{ETR.N1.MappReply,N2,H(ETR.N1.MappReply)}_SK)
       /\ witness(ITR,ETR,N1,N2')
       /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR})
4.State = 3
       /\ Rcv(ITR') =|>
State' := 4 /\ Snd(ITR.ETR.{N2}_SK

end role
role router_ETR (ETR,ITR,MS: agent,
PSK_ItrMs,PSK_MsEtr: symmetric_key,
                              H    : hash_func,
                     Snd, Rcv  : channel(dy),
played_by ETR
def=

local  State         : nat,
SK : symmetric_key
                 N1,N2 : text
   MappRequest,MappReply: text
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id
init State := 0
transition
   1. learn.State = 1
        /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
        /\ witness(ITR,ETR,MS,N1,N2')
        /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})
2.knows.State = 2
        /\ Rcv(ETR') =|>
State' := 3 /\ Snd(ETR.ITR.{ETR.N1.MappReply,N2,H(ETR.N1.MappReply)}_SK)
        /\ witness(ITR,ETR,N1,N2')
        /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR})
```

```
3.cont.State = 3
       /\ Rcv(ITR') =|>
State' := 4 /\ Snd(ITR.ETR.{N2}_SK
end role
role Map_Server (MS,ITR,ETR: agent,
       PSK_ItrMs,PSK_MsEtr: symmetric_key,
                   H    : hash_func,
               Snd, Rcv  : channel(dy),
played_by MS
def=
local ITR,ETR: agent
SK : symmetric_key
                  N1,N2 : text
     MappRequest,MappReply: text,
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id
init State := 0
transition
   1. req1.State = 0 /\ Rcv(start)
     =|>
State' := 1
     /\  Snd(ITR.MS.{ITR.N1.MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_ItrMs)
   2. State = 1
     /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
     /\ witness(ITR,ETR,MS,N1,N2')
     /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})

end role

role session(ITR,ETR,MS: agent,
   PSK_ItrMs,PSK_MsEtr: symmetric_key,
                H    : hash_func,
def=

   SITR,SETR,RITR,RETR,DMS,LMS: channel (dy)
composition
   map_server(MS,ITR,ETR,PSK_ItrMs,PSK_MsEtr,SK,H,DMS,LMS)
   router_itr(ITR,MS,ETR,PSK_ItrMs,PSK_MsEtr,SK,H,SITR,RETR)
   /\ router_etr(ETR,MS,ITR,PSK_ItrMs,PSK_MsEtr,SK,H,SETR,RETR)

end role
role environment()
def=

const itr, etr, ms: agent
       PSK_ItrMS,PSK_MsEtr,SK:symmetric_key
              H   : hash_func,
             N1,N2: protocol_id,


intruder_knowledge = {ITR,MS,ETR,PSK_ItrMs,PSK_MsEtr,H,SK,mapRequest, mapReply}
composition
session(itr,ms,etr, PSK_ItrMs,PSK_MsEtr,SK,h) /\
session(ms,i,PSK_ItrMs,PSK_MsEtr,SK)
session(itr,i,sk,h)/\
session(ETR,i,sk,h)

end role
```

```
goal

%secrecy_of challenge response n1
  security_of secitr, secn1,secms,secetr
  Weak_authentication on itr_ms
  Weak_authentication on itr_ms
  Weak_authentication on itr_ms
  authentication_on etr_itr_n1

end goal
```

APPENDIX –C

Communication Protocol for Internet of Things Version -I

In HLPSL Code

%Macros

%M1 = $Senser_A$, $Senser_B$, N, CH1, exp(G,X)

%M2 = M1, H (SK_A, M1),$Senser_B$,exp(G,X) XOR exp(G,Y)

%M3 = $Senser_B$, $Senser_A$,H(SK_B, $Senser_B$),H(SK_A,exp(G,X) XOR exp(G,Y))

%M4 = $Senser_B$, $Senser_A$, CH1, CH2, exp(G,Y),
H(SK_A,exp(G,X) XOR exp(G,Y)),H(SK_B,$Senser_B$)

%M5 = $Senser_A$, SenserB, CH2, CH3

%M6 = $Senser_B$, SenserA, CH3, CH4

%1. $Senser_A$ -> $Senser_B$: M1,H(SK_A,M1)

%2. $Senser_B$ -> XTR : M2,H(SK_B,M2)

%3. XTR -> $Senser_B$: M3,H(SK_{BXTR}, SK_{AXTR} M3)

%4. $Senser_B$ -> $Senser_A$: M4,H(exp(exp(G,X),Y),M4)

%5. $Senser_A$ -> $Senser_B$: M5,H(exp(exp(G,X),Y),M5)

%6. $Senser_B$ -> $Senser_A$: M6,H(exp(exp(G,X),Y),M6)

```
role sensor_A(
   A,B,XTR : agent,
    SND,RCV    : channel(dy),
    H          : hash_func,
    SKA,SKB    : symmetric_key,
      N,G      : text)

played_by A def=

  local
    State       : nat,
    X,CH1,CH3   : text,
    CH2,CH4     : text,
         GY,Key      : message
         SKAXTR,SKBXTR: symmetric_key

const sec_a_Key : protocol_id

  init  State := 0

  transition

 1. State  = 0 /\ RCV(start) =|>
    State':= 1 /\ X' := new()
            /\ CH1' := new()
            /\ SND(A.B.N.CH1'.exp(G,X').H(SKA.A.B.N.CH1'.exp(G,X')))

 2. State  = 1 /\ RCV(B.A.CH1.CH2'.GY'.
                   H(SKAXTR.xor(exp(G,X),GY')).
                   H(SKAB.B).
                   H(exp(GY',X).B.A.CH1.CH2'.GY'.
                     H(SKA.xor(exp(G,X),GY')).
                     H(B.VGK)))
           =|>
    State':= 2 /\ CH3' := new()
            /\ Key':=exp(GY',X)
            /\ SND(A.B.CH2'.CH3'.H(Key'.A.B.CH2'.CH3'))
            /\ witness(A,B,key1,Key')

 3. State  = 2 /\ RCV(B.A.CH3.CH4'.H(Key.B.A.CH3.CH4')) =|>
    State':= 3 /\ request(A,B,key,Key)
            /\ secret(Key,sec_m_Key,{A,XTR})%XTR must be honest anyway..
end role
```

```
role sensor_B (
   A,B,XTR    : agent,
   SND,RCV    : channel(dy),
   H          : hash_func,
   SKA,SKB    : symmetric_key,
   N,G        : text)

played_by B def=

  local
   State         : nat,
   GX,Key        : message,
   FM1 : hash(symmetric_key.agent.agent.text.text.message),
   FM2 : hash(symmetric_key.agent),
   FM3 : hash(symmetric_key.message),
   M2  : message,
   Y,CH2,CH4      : text,
       CH1,CH3        : text
       SKAXTR,SKBXTR   : symmetric_key

  const sec_b_Key : protocol_id

  init  State := 0

  transition

  1. State = 0 /\ RCV(A.B).N.CH1'.GX'.FM1') =|>
    State':= 1 /\ Y' := new()
           /\ Key':=exp(GX',Y')
           /\ M2' := A.B.N.CH1'.GX'.FM1'.B.xor(GX',exp(G,Y'))
           /\ SND(M2'.H(SKAXTR,SKBXTR.M2'))
           /\ witness(B,A,key,Key')

  2. State = 1 /\ RCV(B.A.FM2'.FM3'.F(SKA.SKB.VGK.MT.FM2'.FM3')) =|>
    State':= 2 /\ CH2' := new()
           /\ SND(  B.A.CH1.CH2'.exp(G,Y).FM3'.FM2'.
                  H(Key.B.A.CH1.CH2'.exp(G,Y).FM3'.FM2'))

  3. State = 2 /\ RCV(A.B.CH2.CH3'.H(Key.A.B.CH2.CH3')) =|>
    State':= 3 /\ CH4' := new()
           /\ SND(B.A.CH3'.CH4'.H(Key.B.A.CH3'.CH4'))
           /\ request(B,A,key1,Key)
           /\ secret(Key,sec_v_Key,{A})

end role

role router_XTR(
   A,B,XTR        : agent,
   SND,RCV        : channel(dy),
   H              : hash_func,
   SKAXTR,SKBXTR  : symmetric_key,
   NIL,G      : text)

played_by XTR def=

  local
   State         : nat,
   GX,GY         : message,
   CH1           : text
  SKA,SKB        : symmetric_key,
```

```
   init
   State := 0

  transition

  1. State = 0  /\ RCV(       A.B.NIL.CH1'.GX'.
                       H(SKA.SKB.MT.VGK.NIL.CH1'.GX').
                           VGK.xor(GX',GY').
                    H(SKAXTR.A.B.N.CH1'.GX'.
                       H(SKBXTR.A.B.N.CH1'.GX').
                           B.xor(GX',GY')))  =|>

     State':= 1 /\ SND(       B.A.H(SKB.B).F(SKA.xor(GX',GY')).
                    F(SKBXTR.B.A.H(SKB.B).F(SKA.xor(GX',GY'))))

end role

role session(
    A,B,XTR : agent,
    H          : hash_func,
    SKA,SKB    : symmetric_key,
    N,G        : text)
def=
    SKAXTR,SKBXTR    : symmetric_key,

  local SND,RCV : channel (dy)

  composition
    sensor_A(A,B,XTR,SND,RCV,H,SKA,SKB,SKAXTR,SKBXTR,N,G)
 /\ router_XTR(A,B,XTR,SND,RCV,H, SKAXTR, SKBXTR, SKB, SKA ,N,G)
 /\ sensor_B(B,A,XTR,SND,RCV, SKA,SKB,SKAXTR,SKBXTR,N,G)

end role


role environment()
def=

  const
    a,b,xtr      : agent,
    h            : hash_func,
    key,key1     : protocol_id,
    sk_a_xtr,sk_b_xtr,ska_skb_i_xtr: symmetric_key,
     n,g         : text

  intruder_knowledge = {a,b,xtr,h,g,n,ska_skb_i_xtr }

  composition
     session(a,b,xtr,h,sk_a_xtr,sk_b_xtr,n,g)
  /\ session(a,b,xtr,h,sk_a_xtr,sk_b_xtr,n,g)

end role

goal
        authentication_on key
  authentication_on key1
  secrecy_of sec_a_Key, sec_b_Key

end goal
```

APPENDIX –D

Security Registration Protocol Final Version

In HLPSL Code

```
role router_ETR (ETR, MS, TKG : agent,
                PUK_ETR,PUK_MS: Public_Key
                     PSK1,PSK2: symmetric_key,
                         H    : hash_func,
                    Snd, Rcv  : channel(dy),

played_by ETR
def=

local  State        : nat,
N1 : text
              M,M2,ACK, : messages
        PVK_ETR, PVK_MS :{text.public_key}_inv(public_key),
EIDPre :messages


init   State := 0

transition

   1. State   = 0  /\ Rcv(start) =|>
State' := 1
                   /\ Snd(TKG.ETR.{PVK_ETR}_PSK1

   2. State   = 1  /\ Rcv(Start) =|>
State' := 2
                   /\ Snd(TKG.MS.{PVK_MS}_PSK2

3.State    = 2  /\ Rcv(PVK_ETR)PSK1) =|>
State'  := 3
                   /\  Snd(ETR.MS.{M,ETR,N1,EIDPre}{PUK_MS}, {H(M,ETR,N1,EIDPre)}{PVK_ETR}
                   /\  witness(ETR,MS,N1)
                   /\  secret (PUK_MS,PVK_ETR,{ETR,MS})
4.State    = 3  /\ Rcv(PVK_MS)PSK2) =|>
     State'  := 4  /\ Snd(MS.ETR.{m2,N1}{PUK_ETR},{H(m2,n1)}{PVK_MS)}
                   /\ witness(MS,ETR,N1)
                   /\ secret (PUK_ETR,PVK_MS,{MS,ETR})

end role
role map_server (MS,ETR: agent,
       PUK_ETR , PUK_MS: Public_Key
              PSK1,PSK2: symmetric_key,
                  H    : hash_func,
             Snd, Rcv  : channel(dy),
played_by MS
def=

local  State        : nat,
N1 : text
              M,M2,ACK, : messages
        PVK_ETR, PVK_MS :{text.public_key}_inv(public_key),
```

```
                EIDPre:messages
init State := 0
transition

1.State    = 2  /\ Rcv(PVK_ETR)PSK1) =|>
State'  := 3
                 /\  Snd(ETR.MS.{M,ETR,N1,EIDPre}{PUK_MS}, {H(M,ETR,N1,EIDPre)}{PVK_ETR}
                 /\  witness(ETR,MS,N1)
                 /\  secret (PUK_MS,PVK_ETR,{ETR,MS})
2.State    = 3  /\ Rcv(PVK_MS)PSK2) =|>
     State'  := 4  /\ Snd(MS.ETR.{m2,N1}{PUK_ETR},{H(m2,n1)}{PVK_MS)}
                 /\ witness(MS,ETR,N1)
                 /\ secret (PUK_ETR,PVK_MS,{MS,ETR})

end role
role the_trusted_ticket_granting (TKG,ETR,MS: agent,
      PUK_ETR , PUK_MS: Public_Key
            PSK1,PSK2: symmetric_key,
                H    : hash_func,
            Snd, Rcv  : channel(dy),
played_by MS
def=

local  State         : nat,
N1 : text
            M,M2,ACK, : messages
       PVK_ETR, PVK_MS :{text.public_key}_inv(public_key),
                EIDPre:messages
init State := 0
transition



 1. State   = 0  /\ Rcv(start) =|>
State' := 1
               /\ Snd(TKG.ETR.{PVK_ETR}_PSK1

 2. State   = 1  /\ Rcv(Start) =|>
State' := 2
               /\ Snd(TKG.MS.{PVK_MS}_PSK2

end role
      role session(ETR,MS,TKG: agent,
     PUK_ETR,PUK_MS,PVK_ETR,PVK_MS: Public_Key
                      PSK1,PSK2: symmetric_key,
H : hash_func,
                            N1: protocol_id
def=

   SETR,SMS,RETR,RMS,DTKG,LTKG: channel (dy)
composition
```

```
   router_ETR(ETR,MS,TKG,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,H,SETR,RETR)
   map_server(MS,ETR,TKG,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,H,SMS,RMS)
   /\the_trusted_ticket_granting(TKG,ETR,MS,PUK_ETR,PUK_MS,PVK_ETR,
   PVK_MS,PSK1,PSK2,H,DTKG,LTKG)

end role
role environment()
def=
const etr,ms,tkg : agent
  PUK_ETR,PUK_MS,PVK_ETR,PVK_MS: Public_Key
                   PSK1,PSK2: symmetric_key,
H : hash_func,
                         N1: protocol_id

intruder_knowledge = {ETR,MS,ETR,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,H,M,M2}
composition
     session(etr,ms,PUK_ETR,PUK_MS,PVK_ETR,PVK_MS,PSK1,PSK2,h) /\
session(etr,i,PVK_ETR,h)/\
session(Ms,i,PVK_MS,h)

end role
goal
  %securecy_of Map_server, Router_ETR on n1
security_of secEtrMS, secMsEtr
  % Router_ETR wakly authenticates Server_MS on _ETR
  weak_authentication_on _ETR
  % Map_server waly authenticates on _MS
  weak_authentication_on_MS
  %Mutual authentication%
  %ROUTER_ETR authenticats Server_MS on n1
authentication on_n1

end goal
```

APPENDIX –E

Resolving Procedure Protocol Final Version

In HLPSL Code

```
role router_ITR (ITR, ETR, MS : agent,
                    PSK_ItrMs,PSK_MsEtr: symmetric_key,
                                 H    : hash_func,
                            Snd, Rcv  : channel(dy),

played_by ITR
def=

local  State         : nat,
SK : symmetric_key
                N1,N2 : text
   MappRequest,MappReply: text
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id

init   State := 0

transition
   1. ask.State = 0 /\ Rcv(start) =|>
State' := 1
       /\  Snd(ITR.MS.{ITR.N1.MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_ItrMs)
   2. learn.State = 1
       /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
       /\ witness(ITR,ETR,MS,N1,N2')
       /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})
3.knows.State = 2
       /\ Rcv(ETR') =|>
State' := 3 /\ Snd(ETR.ITR.{ETR.N1.MappReply,N2,H(ETR.N1.MappReply)}_SK)
       /\ witness(ITR,ETR,N1,N2')
       /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR})
4.State = 3
       /\ Rcv(ITR') =|>
State' := 4 /\ Snd(ITR.ETR.{N2}_SK

end role
role router_ETR (ETR,ITR,MS: agent,
PSK_ItrMs,PSK_MsEtr: symmetric_key,
                              H    : hash_func,
                     Snd, Rcv  : channel(dy),
played_by ETR
def=

local  State         : nat,
SK : symmetric_key
                N1,N2 : text
   MappRequest,MappReply: text
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id
init State := 0
transition
   1. learn.State = 1
        /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
        /\ witness(ITR,ETR,MS,N1,N2')
        /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})
2.knows.State = 2
        /\ Rcv(ETR') =|>
State' := 3 /\ Snd(ETR.ITR.{ETR.N1.MappReply,N2,H(ETR.N1.MappReply)}_SK)
        /\ witness(ITR,ETR,N1,N2')
        /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR})
```

```
3.cont.State = 3
       /\ Rcv(ITR') =|>
State' := 4 /\ Snd(ITR.ETR.{N2}_SK
end role
role Map_Server (MS,ITR,ETR: agent,
       PSK_ItrMs,PSK_MsEtr: symmetric_key,
                   H    : hash_func,
               Snd, Rcv  : channel(dy),
played_by MS
def=
local ITR,ETR: agent
SK : symmetric_key
                  N1,N2 : text
     MappRequest,MappReply: text,
const sec_PSK_ItrMs,PSK_MsEtr,SK :protocol_id
init State := 0
transition
  1. req1.State = 0 /\ Rcv(start)
     =|>
State' := 1
     /\  Snd(ITR.MS.{ITR.N1.MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_ItrMs)
  2. State = 1
     /\ Rcv(MS') =|>
State' := 2 /\ Snd(MS.ETR.{ITR.N1,MappRequest.SK.H(ITR.N1.MappRequest)}_PSK_MsEtr)
     /\ witness(ITR,ETR,MS,N1,N2')
     /\ secret (PSK1,PSK2,sec_SK,{ITR,ETR,MS})

end role
role session(ITR,ETR,MS: agent,
   PSK_ItrMs,PSK_MsEtr: symmetric_key,
               H    : hash_func,
def=

   SITR,SETR,RITR,RETR,DMS,LMS: channel (dy)
composition
   map_server(MS,ITR,ETR,PSK_ItrMs,PSK_MsEtr,SK,H,DMS,LMS)
   router_itr(ITR,MS,ETR,PSK_ItrMs,PSK_MsEtr,SK,H,SITR,RETR)
   /\ router_etr(ETR,MS,ITR,PSK_ItrMs,PSK_MsEtr,SK,H,SETR,RETR)

end role
role environment()
def=

const itr, etr, ms: agent
       PSK_ItrMS,PSK_MsEtr,SK:symmetric_key
              H   : hash_func,
             N1,N2: protocol_id,


intruder_knowledge = {ITR,MS,ETR,PSK_ItrMs,PSK_MsEtr,H,SK,mapRequest, mapReply}
composition
session(itr,ms,etr, PSK_ItrMs,PSK_MsEtr,SK,h) /\
session(ms,i,PSK_ItrMs,PSK_MsEtr,SK)
session(itr,i,sk,h)/\
session(ETR,i,sk,h)

end role
goal

  %secrecy_of PSK_ItrMs,PSK_MsEtr,SK, n1,n2
```

```
security_of secItrMS, secMsEtr, secItrEtr
  % Router_ITR Weak authenticates Map_Server on PSK_ItrMS
  weak_authentication_on PSK_ItrMS
  % Router_ITR Weak authenticates Map_Server on     PSK_MsEtr
  weak_authentication_on PSK_MsEtr
  % Router_ITR Weak authenticates Router_ETR on Sk
  weak_authentication_on PSK_ItrMS on Sk
  %Mutual authentication%
  % ROUTER_ETR authenticats ROUTER_ITR on n1
  authentication_on n1
  % ROUTER_ITR authenticats ROUTER_ETR on n2
  authentication_on n2


end goal
```

APPENDIX- F

Communication Protocol for Internet of Things Final Version

In HLPSL Code

```
%% Macros:
%%  M1: H(PKA.A.B.exp(G, X).exp(UA, Z)).exp(L, Z)
%%  M2: H(PKB.A.B.exp(G, Y ).exp(UB, Z).exp(L, Z)
%% Key: exp(exp(GY, Z), X) = exp(exp(GX, Z), Y )
%%  GX: exp(G, X)
%%  GY: exp(G, Y)
%% 1.Sensor_A -> Sensor_B : xor(exp(G, X), H(PKA.A.B))
%% 2.Sensor_B -> Router_XTR : xor(exp(G, X), H(PKA.A.B)),
%%              xor(exp(G, Y ), H(PKB.A.B))
%% 3.Router_XTR -> sensor_B : xor(exp(GY, Z), M1),
             xor(exp(GX, Z)M2)
%% 4.sensor_B -> sensor_A : xor(exp(GY, Z), M1).H(A.B.Key)
%% 5.sensor_A -> sensor_B : H(A.B.Key)
%% HLPSL:
role sensor_A(A, B, XTR : agent,
            SND, RCV : channel(dy),
                  H : hash func,
                PKA : symmetric key,
                  G : text)
  played_by A
  def=
  local State : nat,
        X, Z : text,
          UA : public_key,
  GY , Key, L : message,
const sec_m_Key : protocol_id,

init State := 0

transition

  1. State = 0 /\RCV(start)= | >
    State’:= 1 /\ X' := new()
             /\ SND(xor(exp(G, X'), H(PKA.A.B)))
  2. State = 1 /\ RCV (xor(exp(GY', Z'),

              H(PKA.A.B.exp(G, X).exp(UA', Z'))).L) = | >
     State’:= 2 /\ Key’ := exp(exp(GY’,Z’),X)
              /\ SND(H(A.B.Key'))
              /\ witness(A, B, key1, Key')

  3.  State = 2  /\ RCV (A.B.Key) = | >
      State’:= 3 /\ request(A, B, key, Key)
               /\ secret(Key, sec_m_Key, A, B)

end role
role sensor_B (A, B, XTR : agent,
             SND, RCV : channel(dy),
                   H : hash func,
             PKA, PKB : symmetric key,
                   G :text)
played by B
def=
 local State : nat,
       X, Y, Z : text,
        GX, GY : message,
             UB : public key,
            Key : message,
             M1: hash(symmetric-key.agent.agent.message.message).message,
```

```
              M2: hash(symmetric-key.agent.agent.message.message).message

const sec v Key : protocol id

init State := 0

transition
   1. State = 0   /\ RCV (xor(exp(G, X'), H(PKA.A.B))) = | >
      State':= 1  /\ Y':= new()
                 /\ SND(xor(exp(G, X'),
                    H(PKA.A.B)).xor(exp(G, Y'), H(PKB.A.B)))

   2. State = 1   /\  RCV (xor(exp(GY, Z'),M1').
                     xor(exp(GX', Z'), M2')) = |>
      State':= 2  /\  SND(xor(exp(GY, Z'), M1))
   3. State = 2   /\  RCV (H(A.B.exp(exp(GX',Z'), Y )))= |>
      State':= 3  /\  Key':= exp(exp(GX',Z'),Y )
                    /\  SND(H(A.B.Key'))
                    /\  request(B, A, key1, Key)
                    /\  secret(Key, sec_v_Key, B,A)
                    /\  witness(B,A,key,Key')
end role
role router_xtr(A, B, XTR : agent,
                SND, RCV : channel(dy),
                       H : hash func,
                PKA, PKB : symmetric key,
                        G : text)
played by XTR
def=
 local State : nat,
     X, Y, Z : text,
      UA, UB : public key,
      GX, GY : message
  init State := 0
  transition
  1. State = 0 /\ RCV (xor(exp(G, X'),
            H(PKA.A.B)).xor(exp(G,Y'), H(PKB.A.B)))= |>
   State':= 1  /\ Z' := new()
            /\ UA':= new()
            /\ UB':= new()
            /\ GY':= new()
            /\ GX':= new()
            /\ SND(xor(exp(GY', Z'),
          H(PkA.A.B.exp(G, X').exp(UA',Z')).
          exp(G, Z')).
          xor(exp(GX',Z'), H(PKB.A.B.
          exp(G, Y').exp(UB', Z')).exp(G, Z')))
end role
role session( A, B, XTR : agent,
                     H : hash func,
              PKA, PKB : symmetric key,
                UA, UB :public key,
                     G : text)
def=
local SND, RCV : channel (dy)
composition
    sensor_a(A,B,XTR,SND,RCV,H,PKA,G)
 /\ sensor_b(A,B,XTR,SND,RCV,H,PKA,PKB,G)
 /\ router_xtr(A,B,XTR,SND,RCV,H,PKA,PKB,G)
end role
```

```
role environment()
def=
const a, b, xtr : agent,
             h : hash func,
      key, key1 : protocol_id,
     pa, pb, pi :symmetric_key,
     ua, ub, ui :public key,
             g : text
 intruder knowledge = {a, b, xtr, g, h, pi, ua, ub, ui}
 composition
    session(b, a, xtr, h, pa, pb, ua, ub, g)
 /\ session(i, b, xtr, h, pi, pb, ui, ub, g)
 /\ session(a, i, xtr, h, pa, pi, ua, ui, g)
end role

goal
authentication on key
authentication on key1
secrecy-of sec-m-Key, sec-v-Key
end goal
```

APPENDIX- G

Security Refinement protocols for Internet of Things

In HLPSL Code

```
role sensor_A (A, B, ITR,ETR : agent,
                  SND, RCV : channel(dy),
                  H : hash func,
                 PKA : symmetric key,
                 G : text)

    played_by A
    def=

    local State : nat,
          X, Z : text,
          UA : public_key,
          GY , Key, L : message,

   const sec_m_Key : protocol_id,

init State := 0

  transition

    16. State = 0 /\RCV(start)= | >
     State':= 1 /\ X' := new()
           /\ SND(xor(exp(G, X'), H(PKA.A.B)))
    17. State = 1 /\ RCV (xor(exp(GY', Z'),
              H(PKA.A.B.exp(G, X).exp(UA', Z'))).L) = | >
      State':= 2 /\ Key' := exp(exp(GY',Z'),X)
            /\ SND(H(A.B.Key'))
            /\ witness(A, B, key1, Key')
    18.  State = 2  /\ RCV (A.B.Key) = | >
       State':= 3 /\ request(A, B, key, Key)
             /\ secret(Key, sec_m_Key, A, B)
     end role

    role sensor_B (A, B, ITR,ETR : agent,
                    SND, RCV : channel(dy),
                    H : hash func,
                    PKA, PKB : symmetric key,
                   G :text)
   played by B
   def=

    local State : nat,
        X, Y, Z : text,
         GX, GY : message,
             UB : public key,
            Key : message,
             M1: hash(symmetric-key.agent.agent.message.message).message,
             M2: hash(symmetric-key.agent.agent.message.message).message
   const sec v Key : protocol id
   init State := 0

   transition

    16. State = 0   /\ RCV (xor(exp(G, X'), H(PKA.A.B))) = | >
       State':= 1  /\ Y':= new()
              /\ SND(xor(exp(G, X'),
                H(PKA.A.B)).xor(exp(G, Y'), H(PKB.A.B)))

     17. State = 1   /\  RCV (xor(exp(GY, Z'),M1').
```

```
                 xor(exp(GX', Z'), M2')) = |>
       State':= 2  /\  SND(xor(exp(GY, Z'), M1))
     18. State = 2   /\  RCV (H(A.B.exp(exp(GX',Z'), Y )))= |>
       State':= 3  /\  Key':= exp(exp(GX',Z'),Y )
                /\  SND(H(A.B.Key'))
                /\  request(B, A, key1, Key)
                /\  secret(Key, sec_v_Key, B,A)
                /\  witness(B,A,key,Key')
   end role

  role router_ETR (ETR, MS, TKG,ITR, A,B, : agent,
                 SND, RCV : channel(dy),
                 H: hash func,
                PSK1, PSK2, PSK3, PSK4: symmetric key
                PKA, PKB: symmetric key
                      G: text

        played_by ETR
 def=

 local State: nat,
 N1, N2, N3,N4: text
 X, Y, Z: text
        M, M2, ACK: messages
        UA, UB, PK_ETR, PK_MS: Public key
        K_ETR, K_MS: {text.public_key}_ inv(public_key),
 SK: symmetric
 EIDPre: messages
 GY, Key, L: message,
        init   State := 0

   transition

    1. State   = 0  /\ Rcv(start) =|>
State' := 1
           /\ Snd(TKG.ETR.{K_ETR}_PSK1

   2. State   = 1  /\ Rcv(Start) =|>
State' := 2
             /\ Snd(TKG.MS.{PK_MS}_PSK2

   3. State    = 2 /\ Rcv(K_ETR)PSK1) =|>
State'  := 3 /\  Snd(ETR.MS.{M,ETR,N1,EIDPre}{PK_MS}, {H(M,ETR,N1,EIDPre)}{K_ETR}
             /\  witness(ETR,MS,N1)
             /\  secret (PK_MS,K_ETR,{ETR,MS})
   13. learn.State = 3
                 /\ Rcv(MS') =|>
       State' := 4 /\ Snd(MS.ETR.{ITR.N3,MappRequest.SK.H(ITR.N3.MappRequest)}_PSK3)
                 /\ witness(ITR,ETR,MS,N3,N4')
                 /\ secret (PSK3,PSK4,sec_SK,{ITR,ETR,MS})
   14.knows.State = 4
                 /\ Rcv(ETR') =|>
       State' := 5 /\ Snd(ETR.ITR.{ETR.N1.MappReply,N4,H(ETR.N3.MappReply)}_SK)
                 /\ witness(ITR,ETR,N3,N4')
                 /\ secret (PSK3,PSK4,sec_SK,{ITR,ETR})
   15.cont.State = 5
                 /\ Rcv(ITR') =|>
State' := 6 /\ Snd(ITR.ETR.{N3}_SK

    18. State = 0 /\ RCV (xor(exp(G, X'),
```

```
                H(PKA.A.B)).xor(exp(G,Y'), H(PKB.A.B)))= |>
        State':= 1    /\ Z' := new()
                      /\ UA':= new()
                      /\ UB':= new()
                      /\ GY':= new()
                      /\ GX':= new()
                      /\ SND(xor(exp(GY', Z'),
              H(PkA.A.B.exp(G, X').exp(UA',Z')).
              exp(G, Z')).
              xor(exp(GX',Z'), H(PKB.A.B.
              exp(G, Y').exp(UB', Z')).exp(G, Z')))

      end role
role router_ITR (ETR, MS, A,B,TKG : agent,
                SND, RCV : channel(dy),
                H: hash func,
               PSK1,PSK2,PSK3,PSK4 : symmetric key
               PKA, PKB: symmetric key
                     G: text
         played_by ITR
    def=

    local State: nat,
    N3,N4: text
    X, Y, Z :text
         M,M2,ACK :messages
         UA, UB, PK_ETR,PK_MS: Public key
         K_ETR, K_MS: {text.public_key}_ inv(public_key),
    SK: symmetric
    EIDPre: messages
    GY, Key, L: message,
         init   State := 0

transition

 6. ask.State = 0 /\ Rcv(start) =|>
                 State' := 1
                    /\  Snd(ITR.MS.{ITR.N3.MappRequest.SK.H(ITR.N3.MappRequest)}_PSK3)
 7. learn.State = 1
                    /\ Rcv(MS') =|>
       State' := 2 /\ Snd(MS.ETR.{ITR.N3,MappRequest.SK.H(ITR.N3.MappRequest)}_PSK4)
                    /\ witness(ITR,ETR,MS,N3,N4')
                    /\ secret (PSK3,PSK4,sec_SK,{ITR,ETR,MS})
 8.knows.State = 2
                    /\ Rcv(ETR') =|>
       State' := 3 /\ Snd(ETR.ITR.{ETR.N3.MappReply,N4,H(ETR.N3.MappReply)}_SK)
                    /\ witness(ITR,ETR,N3,N4')
                    /\ secret (PSK3,PSK4,sec_SK,{ITR,ETR})
  9.State = 3  /\ Rcv(ITR') =|>
       State' := 4 /\ Snd(ITR.ETR.{N4}_SK

  18.State = 0 /\ RCV (xor(exp(G, X'),
                 H(PKA.A.B)).xor(exp(G,Y'), H(PKB.A.B)))= |>
       State':= 1  /\ Z' := new()
                /\ UA':= new()
                /\ UB':= new()
                /\ GY':= new()
                /\ GX':= new()
                /\ SND(xor(exp(GY', Z'),
              H(PkA.A.B.exp(G, X').exp(UA',Z')).
```

```
              exp(G, Z')).
              xor(exp(GX',Z'), H(PKB.A.B.
              exp(G, Y').exp(UB', Z')).exp(G, Z')))

end role
role map_server (MS,ETR,ITR: agent,
               PSK1,PSK2,PSK3,PSK4 : symmetric key
                 SND, RCV : channel(dy),
          H: hash func,
               played_by MS
    def=
    local State: nat,
    N1, N2,N3,N4: text
       M,M2,ACK :messages
       PK_ETR,PK_MS: Public key
       K_ETR, K_MS: {text.public_key}_ inv(public_key),
SK: symmetric
EIDPre: messages
       init   State := 0
   4. State    = 2    /\ Rcv(K_ETR)PSK1) =|>
 State'  := 3    /\  Snd(ETR.MS.{M,ETR,N1,EIDPre}{PK_MS}, {H(M,ETR,N1,EIDPre)}{K_ETR}
                 /\ witness(ETR,MS,N1)
                 /\  secret (PK_MS,K_ETR,{ETR,MS})
   5.State    = 3     /\ Rcv(K_MS)PSK2) =|>
   State'  := 4 /\ Snd(MS.ETR.{m2,N1}{PK_ETR},{H(m2,n1)}{K_MS)}
                 /\ witness(MS,ETR,N1)
                 /\ secret (PK_ETR,K_MS,{MS,ETR})
  10. req1.State = 0  /\ Rcv(start)
                   =|>
       State' := 1
                     /\  Snd(ITR.MS.{ITR.N3.MappRequest.SK.H(ITR.N3.MappRequest)}_PSK3)
  12. State = 1
                    /\ Rcv(MS') =|>
       State' := 2 /\ Snd(MS.ETR.{ITR.N3,MappRequest.SK.H(ITR.N3.MappRequest)}_PSK4)
                    /\ witness(ITR,ETR,MS,N3,N4')
                    /\ secret (PSK3,PSK4,sec_SK,{ITR,ETR,MS})


      end role

role the_trusted_ticket_granting (TKG,ETR,MS: agent,
                 PUK_ETR , PUK_MS: Public_Key
                 PSK1,PSK2: symmetric_key,
                          H    : hash_func,
                   Snd, Rcv  : channel(dy),
played_by TKG
def=

local  State         : nat,
N1 : text
              M,M2,ACK, : messages
              PVK_ETR, PVK_MS :{text.public_key}_inv(public_key),
              EIDPre:messages
init State := 0

transition

 1. State   = 0  /\ Rcv(start) =|>
State' := 1
                  /\ Snd(TKG.ETR.{PVK_ETR}_PSK1
```

```
 2. State   = 1  /\ Rcv(Start) =|>
State' := 2
                 /\ Snd(TKG.MS.{PVK_MS}_PSK2
     end role

     role session(ETR,ITR,A,B,MS,TKG: agent,
              PK_ETR, PK_MS, K_ETR, K_MS, UA, UB: Public_Key
              PSK1, PSK2, PSK3, PSK4, PKA, PKB, SK: symmetric_key,
      H: hash func,
                  G: text)
   def=
    local SND, RCV : channel (dy
      composition
          sensor_a (A, B, ITR, ETR, SND, RCV, H, PKA, G)
        /\ sensor_b(A,B,ITR,ETR,SND,RCV,H,PKA,PKB,G)
   /\ router_itr(A,B,ITR,ETR,MS,SND,RCV,H,PSK3,PSK4,SK,PKA,PKB,G)
   /\ router_etr(A,B,ITR,ETR,MS,TKG,SND,RCV, PK_ETR,PK_MS,K_ETR,K_MS
H,PSK1,PSK2,PSK3,PSK4,SK,PKA,PKB,G)
   /\ map_server(MS,ITR,ETR,TKG,PK_ETR,PK_MS,K_ETR,K_MS,PSK1,PSK2,PSk3,PSK4)
   /\the_trusted_ticket_granting(TKG,ETR,MS,PK_ETR,PK_MS,K_ETR, K_MS,PSK1,PSK2,H)

end role

    role environment()
    def=
    const a, b, itr,etr,ms,tkg : agent,
             h : hash func,
    key, key1 : protocol_id,
       pa, pb, psk1,pak2,psk3,psk4,sk,pi :symmetric_key,
      PK_ETR,PK_MS,K_ETR,K_MS ua, ub, ui :public key,
             g : text

    intruder knowledge = {a, b, itr,etr,ms,tkg, g, h, pi, ua, ub, ui ,pk_etr,pk_ms,k_etr,k_ms,
h,psk1,psk2,psk3,psk4,m,m2 sk,maprequest, mapreply}
     composition
      session(b, a, itr,etr h, pa, pb, ua, ub, g)
     /\ session(i, b,etr, itr, h, pi, pb, ui, ub, g)
     /\ session(a, i, itr,etr, h, pa, pi, ua, ui, g)
     /\ session(etr,i,k_etr,h)
      /\ session(ms,i,k_ms,h)
     /\ session(itr,ms,etr, psk3,psk4,sk,h)
      /\   session(ms,i,psk3,psk4,sk)
      /\   session(itr,i,sk,h)/\
      /\  session(etr,i,sk,h)
end role

goal
       %securecy_of Map_server, Router_ETR Router_
        ITR Sensor_A,Sensor_B on psk1,psk2,psk3,psk4,n1,n2,n3
 security_of secpsk1, secpsk2,secpsk3,secpsk4,sec-m-Key,sec-v-Key
       % Router_ETR wakly authenticates Server_MS on _ETR
        weak_authentication_on _ETR
 % Router_ITR wakly authenticates Map_Server on  psk3
 weak_authentication_on _ETR
 weak_authentication_on psk3
        % Router_ITR wakly authenticates Router_ETR on Sk
        weak_ authenticates _on psk3 on sk
        % Map_server waly authenticates on _MS
   weak_ authenticates _on_MS
   %Mutual authentication%
```

```
%ROUTER_ETR authenticates Server_MS on n1
authentication on_n1
         % ROUTER_ETR authenticats ROUTER_ITR on n3
        authentication_on n3
        % ROUTER_ITR authenticates ROUTER_ETR on n4
           authentication_on n4
          sensor_a authenticates aensor_b on sk

 end goal
```